

ASSIGNMENT-03

Build CNN Model for Classification Of Flowers

Assignment Date	05-October 2022
Student Name	Dhanalakshmi V
Student Roll Number	922519205030
Maximum Marks	2 Marks

QUESTION 1:

Download the Dataset

Dataset is downloaded and uploaded

QUESTION 2:

Image Augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
rose_datagen=ImageDataGenerator(rescale=1.255,horizontal_flip=True,vertical_flip=True)
x_data= rose_datagen.flow_from_directory(r"/content/drive/MyDrive/IBM Assignments/flowers",target_size=(64,64),class_mode="categorical",batch_size=24)
x_data.class_indices
```



The screenshot shows a Jupyter Notebook interface with four code cells. The first cell imports ImageDataGenerator. The second cell creates an ImageDataGenerator object with rescale=1.255, horizontal_flip=True, and vertical_flip=True. The third cell uses flow_from_directory to load images from a specific path, setting target_size=(64,64) and class_mode="categorical". The output of this cell is "Found 4317 images belonging to 5 classes." The fourth cell prints x_data.class_indices, showing a dictionary mapping flower names to indices: {'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}.

```
[5] from tensorflow.keras.preprocessing.image import ImageDataGenerator

[7] rose_datagen=ImageDataGenerator(rescale=1.255,horizontal_flip=True,vertical_flip=True)

[8] x_data= rose_datagen.flow_from_directory(r"/content/drive/MyDrive/IBM Assignments/flowers",target_size=(64,64),
Found 4317 images belonging to 5 classes.

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

QUESTION 3:

Create Model

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

✓
0s



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

QUESTION 4:

Add Layers (Convolution, MaxPooling, Flatten, Dense-(Hidden Layers), Output

```
model.add(Convolution2D(34, (3, 3), activation="relu", strides=(1, 1), input_shape=(64, 64, 3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.summary()
model.add(Dense(200, activation="relu"))
model.add(Dense(200, activation="relu"))
model.add(Dense(5, activation="softmax"))
```

✓
1s

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

✓
0s

```
[19] model = Sequential()
```

✓
0s

```
[20] model.add(Convolution2D(34, (3, 3), activation="relu", strides=(1, 1), input_shape=(64, 64, 3)))
```

✓
0s

```
[21] model.add(MaxPooling2D(pool_size=(2, 2)))
```

✓
0s

```
[22] model.add(Flatten())
```

✓
0s

```
[23] model.summary()
```

Model: "sequential_2"

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 62, 62, 34)	952
max_pooling2d_1 (MaxPooling 2D)	(None, 31, 31, 34)	0
flatten_1 (Flatten)	(None, 32674)	0

=====
Total params: 952
Trainable params: 952
Non-trainable params: 0
=====

✓
3s

```
[24] model.add(Dense(200, activation="relu"))
     model.add(Dense(200, activation="relu"))
```

✓
0s

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



QUESTION 5:

Compile the Model

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

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

QUESTION 6:

Fit the model

```
model.fit(x_data, epochs= 5, steps_per_epoch= len(x_data) ,validation_data=0.0,validation_steps=0.0)
```

```
✓ model.fit(x_data, epochs= 5, steps_per_epoch= len(x_data) ,validation_data=0.0,validation_steps=0.0)
```

Epoch 1/5
90/90 [=====] - 34s 381ms/step - loss: 0.6416 - accuracy: 0.8110
Epoch 2/5
90/90 [=====] - 34s 374ms/step - loss: 0.4987 - accuracy: 0.8425
Epoch 3/5
90/90 [=====] - 35s 382ms/step - loss: 0.5009 - accuracy: 0.8443
Epoch 4/5
90/90 [=====] - 34s 375ms/step - loss: 0.4476 - accuracy: 0.8647
Epoch 5/5
90/90 [=====] - 36s 400ms/step - loss: 0.4143 - accuracy: 0.8663
<keras.callbacks.History at 0x7fab0c8f2690>

QUESTION 7:

Save the Model

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

```
✓ model.save('flowers.h5')
```

QUESTION 8:

Test the Model

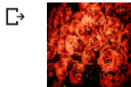
```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
img = image.load_img(r"/content/drive/MyDrive/IBM Assignments/flowers/rose/10090824183_d02c613f10_m.jpg")
img
```

```
✓ [95] import numpy as np
0s      from tensorflow.keras.models import load_model
      from tensorflow.keras.preprocessing import image
```

```
✓ [96] model = load_model('flowers.h5')
0s
```

```
✓ [99] .mg(r"/content/drive/MyDrive/IBM Assignments/flowers/rose/10503217854_e66a804309.jpg",target_size=(64,64))
0s
```

```
✓ [100] img
0s
```



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

```
✓ [101] x = image.img_to_array(img)
s
```

```
✓ [102] x
s
```

```
array([[ 0.,  2.,  0.],
       [ 0.,  2.,  0.],
       [ 0.,  2.,  0.],
       ...,
       [ 92., 14.,  0.],
       [ 61., 13.,  9.],
       [ 17.,  7.,  5.]],

       [[ 0.,  2.,  0.],
       [ 0.,  2.,  0.],
       [ 0.,  2.,  0.],
       ...,
       [150.,  3.,  0.],
       [ 85., 10.,  7.]])
```

```
x=np.expand_dims(x,axis=0)
pred= model.predict(x)
```

```
✓ [103] x=np.expand_dims(x,axis=0)
0s
```

```
✓ [104] pred=model.predict(x)
0s
```

```
1/1 [=====] - 0s 59ms/step
```

```
x_data.class_indices
```

```
✓ [106] x_data.class_indices
```

0s

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

✓

0s

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

#predicting



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

```
'rose'
```