

### ASSIGNMENT – 3

Assignment Date	06- October 2022
Student Name	P.Nithya Sri
Student Roll No.	510919205026
Maximum marks	2 marks

*PROBLEM STATEMENT: Build CNN  
Model for Classification of  
Flowers.*

#### QUESTION – 1:

## DOWNLOAD THE DATASET

Q

{x}

▼ Loading Dataset

```
!unzip 'M:\software\AI_TRAINING_IBM\Flowers-Dataset.zip'
```

unzip: cannot find or open M:\software\AI\_TRAINING\_IBM\Flowers-Dataset.zip, M:\software\AI\_TRAINING\_IBM\Flowers-Dataset.zip.zip or M:\software\AI\_TRAINING\_IBM\Flowers-Dataset.zip

## QUESTION – 2:

### DATA/ IMAGE AUGMENTATION

▼ Data Augmentation

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

## QUESTION – 3:

### TRAINING & TESTING

▼ Training and Testing

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

```
In [9]: xtrain=train_datagen.flow_from_directory('M:\\software\\AI_TRAINING_IBM\\flowers',class_mode='categorical',target_size=(64,64),batch_size=100)
```

Found 4317 images belonging to 5 classes.

```
In [10]: xtest=test_datagen.flow_from_directory('M:\\software\\AI_TRAINING_IBM\\flowers',class_mode='categorical',target_size=(64,64),batch_size=100)
```

Found 4317 images belonging to 5 classes.

## QUESTION

- 4 &

## QUESTION 5:

CREATE

MODEL:

ADD LAYERS

- ▼ Importing the models and the layers

```
✓ [7] from tensorflow.keras.models import Sequential  
0s from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
```

```
✓ 1s ▶ model=Sequential()  
model.add(Convolution2D(64,(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'))  
model.add(Dense(5,activation='softmax'))
```

## QUESTION – 6:

COMPILE THE MODEL:

- ▼ Compile

```
✓ 0s ▶ model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

## QUESTION – 7:

FIT THE MODEL:

## Fit the model.

```
model.fit_generator(xtrain,
                    steps_per_epoch=len(xtrain),
                    epochs=20,
                    validation_data=xtest,
                    validation_steps=len(xtest))
```

C:\Users\mm\AppData\Local\Temp\ipykernel\_6696\312721451.py:1: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.  
model.fit\_generator(xtrain,

Epoch 1/20

44/44 [=====] - 37s 835ms/step - loss: 1.9038 - accuracy: 0.3836 - Val loss: 1.1672 - Val accuracy: 0.5219 Epoch 2/20

44/44 [=====] - 34s 779ms/step - loss: 1.0908 - accuracy: 0.5606 - Val loss: 1.0398 - Val accuracy: 0.5965 Epoch 3/20

44/44 [=====] - 36s 815ms/step - loss: 1.0262 - accuracy: 0.5925 - Val loss: 1.0038 - Val accuracy: 0.6185 Epoch 4/20

44/44 [=====] - 36s 823ms/step - loss: 0.9335 - accuracy: 0.6410 - Val loss: 0.8923 - Val accuracy: 0.6560 Epoch 5/20

44/44 [=====] - 36s 809ms/step - loss: 0.8781 - accuracy: 0.6604 - Val loss: 0.8886 - Val accuracy: 0.6646 Epoch 6/20

44/44 [=====] - 34s 764ms/step - loss: 0.8512 - accuracy: 0.6713 - Val loss: 0.8784 - Val accuracy: 0.6771 Epoch 7/20

44/44 [=====] - 33s 758ms/step - loss: 0.7922 - accuracy: 0.6931 - Val loss: 0.7586 - Val accuracy: 0.7121 Epoch 8/20

44/44 [=====] - 35s 811ms/step - loss: 0.7471 - accuracy: 0.7107 - Val loss: 0.6955 - Val accuracy: 0.7262 Epoch 9/20

44/44 [=====]	- 35s 795ms/step	- loss:	0.7157	- ac
curacy: 0.7311 - Val loss: 0.6671 - Val	accuracy: 0.7482			
Epoch 10/20				
44/44 [=====]	- 36s 817ms/step	- loss:	0.6867	- ac
curacy: 0.7336 - Val loss: 0.6537 - Val	accuracy: 0.7524			
Epoch 11/20				
44/44 [=====]	- 37s 851ms/step	- loss:	0.6314	- ac
curacy: 0.7628 - Val loss: 0.6081 - Val	accuracy: 0.7751			

Epoch 12/20				
44/44 [=====] - 34s 773ms/step curacy: 0.7744 - Val loss: 0.6052 - Val	accuracy: 0.7716	- loss:	0.6109	- ac
Epoch 13/20				
44/44 [=====] - 34s 777ms/step curacy: 0.7853 - Val loss: 0.5747 - Val	accuracy: 0.7760	- loss:	0.5710	- ac
Epoch 14/20				
44/44 [=====] - 33s 763ms/step curacy: 0.7924 - Val loss: 0.4951 - Val	accuracy: 0.8112	- loss:	0.5516	- ac
Epoch 15/20				
44/44 [=====] - 34s 769ms/step curacy: 0.8019 - Val loss: 0.4531 - Val	accuracy: 0.8334	- loss:	0.5265	- ac
Epoch 16/20				
44/44 [=====] - 32s 721ms/step curacy: 0.8177 - Val loss: 0.3755 - Val	accuracy: 0.8631	- loss:	0.4957	- ac
Epoch 17/20				
44/44 [=====] - 32s 739ms/step curacy: 0.8272 - Val loss: 0.5578 - Val	accuracy: 0.7797	- loss:	0.4737	- ac
Epoch 18/20				
44/44 [=====] - 30s 680ms/step curacy: 0.8274 - Val loss: 0.3953 - Val	accuracy: 0.8511	- loss:	0.4653	- ac
Epoch 19/20				
44/44 [=====] - 25s 578ms/step curacy: 0.8395 - Val loss: 0.3990 - Val	accuracy: 0.8550	- loss:	0.4252	- ac
Epoch 20/20				
44/44 [=====] - 26s 597ms/step curacy: 0.8529 - Val loss: 0.3112 - Val	accuracy: 0.8888	- loss:	0.3946	- ac

Out[19]:

<Keras.callbacks.History at 0x2b10b08c370>

## **QUESTION – 8:**

### **SAVING THE**

### **MODEL**

{x}

## Saving Model



```
✓ [11] model.save('Flower.h5')  
0s
```

✓  
0s



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

## QUESTION – 9:

### TEST THE MODEL

#### Testing the model

```
In [22]: img=image.load_img('M:\\software\\AI_TRAINING_IBM\\flowers\\sunflower\\6953297_8576bf4ea3.jpg',target_size=(64,64))  
x=image.img_to_array(img)  
x=np.expand_dims(x,axis=0)  
prediction=np.argmax(model.predict(x))  
op=['daisy','dandelion','rose','sunflower','tulip']  
op[prediction]
```

```
1/1 [=====] - 0s 22ms/step
```

```
Out[22]: 'sunflower'
```

## QUESTION -10:

## TESTING THE MODEL

```
In [24]: img=image.load_img('M:\\software\\AI_TRAINING_IBM\\download.jpg',target_size=(64,64))#randomly downloaded testing
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
prediction=np.argmax(model.predict(x))
op=['daisy','dandelion','rose','sunflower','tulip']
op[prediction]
```

```
1/1 [=====] - 0s 22ms/step
```

```
Out[24]: 'sunflower'
```

```
In [ ]:
```

---