

ASSIGNMENT-03

Build CNN Model for Classification Of Flowers

Assignment Date	5 October 2022
Student Name	SANJAY V
Student Roll Number	113219071036
Maximum Marks	2 Marks

QUESTION 1:

Download the Dataset

Dataset is downloaded and uploaded

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf

from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense

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

import cv2
```

QUESTION 2:

Image Augmentation

```
data_path = '/content/drive/MyDrive/CNN/flowers'
batch_size = 32
target_size = (64, 64)
```

[illegible]

```
subset="training",
class_mode = 'categorical')
```

```
X_test = test_datagen.flow_from_directory(data_path,
                                         target_size=target_size,
                                         batch_size=batch_size,
                                         subset="validation",
                                         class_mode='categorical')
```

```
[2] data_path = '/content/drive/MyDrive/CNN/flowers'
     batch_size = 32
     target_size = (64, 64)

[3] train_datagen = ImageDataGenerator(rescale=1./255,
                                       shear_range=0.2,
                                       zoom_range=0.2,
                                       width_shift_range=0.1,
                                       height_shift_range=0.1,
                                       horizontal_flip=True,
                                       validation_split=0.2)

test_datagen = ImageDataGenerator(rescale=1. / 255, validation_split=0.2)

[4] X_train = train_datagen.flow_from_directory(data_path,
                                              target_size=target_size,
                                              batch_size=batch_size,
                                              subset="training",
                                              class_mode = 'categorical')

X_test = test_datagen.flow_from_directory(data_path,
                                          target_size=target_size,
                                          batch_size=batch_size,
                                          subset="validation",
                                          class_mode='categorical')

Found 16 images belonging to 5 classes.
Found 4 images belonging to 5 classes.
```

QUESTION 3:

Create Model

```
model = Sequential()
```

```
+ Code + Text

[5] model = Sequential()
```

QUESTION 4:

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

Output

```
model.add(Convolution2D(32, (3, 3), input_shape=(64, 64, 3),
activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

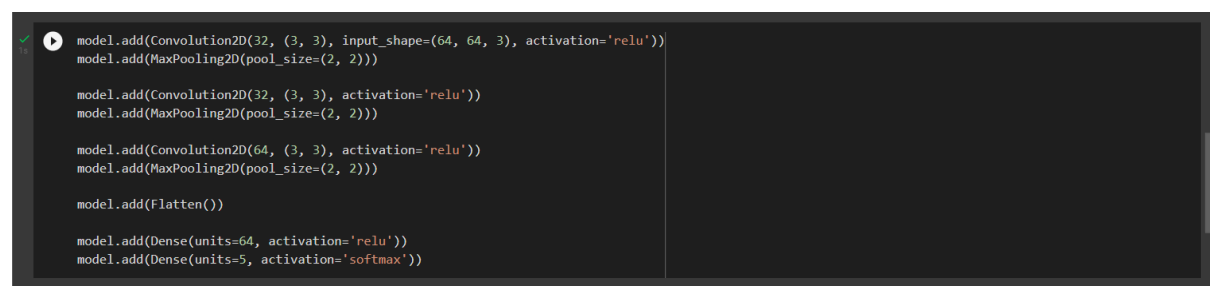
model.add(Convolution2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=5, activation='softmax'))

model.summary()
```



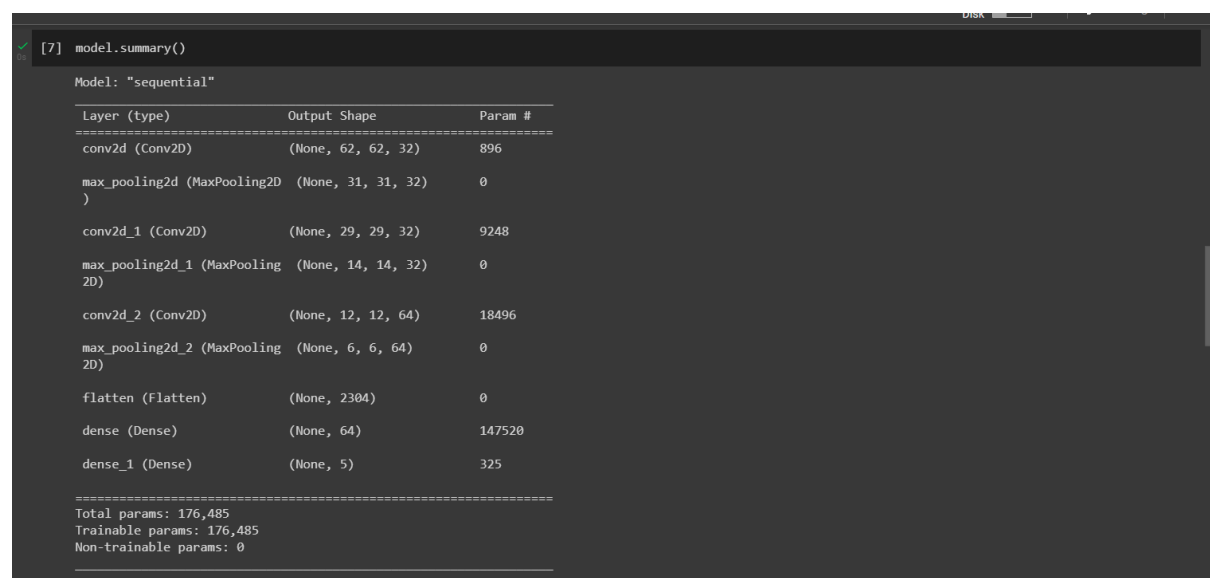
```
model.add(Convolution2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Convolution2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=5, activation='softmax'))
```



```
[7] model.summary()

Model: "sequential"

```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_1 (Conv2D)	(None, 29, 29, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	0
conv2d_2 (Conv2D)	(None, 12, 12, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 64)	0
flatten (Flatten)	(None, 2304)	0
dense (Dense)	(None, 64)	147520
dense_1 (Dense)	(None, 5)	325

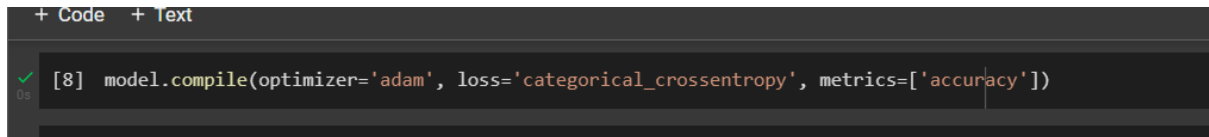
```

Total params: 176,485
Trainable params: 176,485
Non-trainable params: 0
```

QUESTION 5:

Compile the Model

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

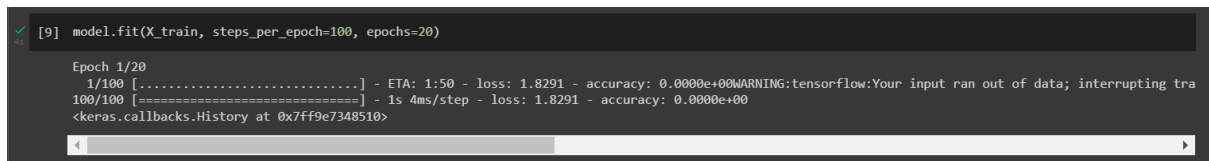


A screenshot of a Jupyter Notebook cell. At the top, there are tabs for '+ Code' and '+ Text'. Below the tabs, a green checkmark icon and '0s' are visible. The code cell contains the following line: `[8] model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])`. The code is highlighted in a dark background.

QUESTION 6:

Fit the model

```
model.fit(X_train, steps_per_epoch=100, epochs=20)
```

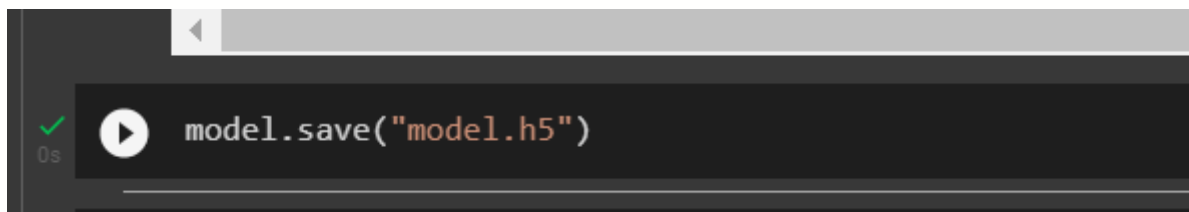


A screenshot of a Jupyter Notebook cell. At the top, a green checkmark icon and '0s' are visible. The code cell contains the following line: `[9] model.fit(X_train, steps_per_epoch=100, epochs=20)`. Below the code, there is a progress bar and some output text:
Epoch 1/20
1/100 [.....] - ETA: 1:50 - loss: 1.8291 - accuracy: 0.0000e+00
100/100 [=====] - 1s 4ms/step - loss: 1.8291 - accuracy: 0.0000e+00
<keras.callbacks.History at 0x7ff9e7348510>

QUESTION 7:

Save the Model

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



A screenshot of a Jupyter Notebook cell. At the top, there is a play button icon and '0s'. The code cell contains the following line: `model.save("model.h5")`. The code is highlighted in a dark background.

QUESTION 8:

Test the Model

```
def predict():  
    img =  
    image.load_img("/content/drive/MyDrive/CNN/flowers/rose/118974357_0faa23cce  
9_n.jpg", target_size=target_size)  
    x = image.img_to_array(img)  
    x = tf.expand_dims(x,0)  
  
    labels = ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']  
  
    pred = model.predict(x)  
    prediction = labels[np.argmax(pred[0])]
```

```

print(f'The given image is a {prediction}')

plt.imshow(plt.imread("/content/drive/MyDrive/CNN/flowers/rose/118974357_0f
aa23cce9_n.jpg"))
plt.axis('off')
plt.show()

predict()

```

```

[11] def predict():
    img = image.load_img("/content/drive/MyDrive/CNN/flowers/rose/118974357_0faa23cce9_n.jpg", target_size=target_size)
    x = image.img_to_array(img)
    x = tf.expand_dims(x,0)

    labels = ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']

    pred = model.predict(x)
    prediction = labels[np.argmax(pred[0])]

    print(f'The given image is a {prediction}')
    plt.imshow(plt.imread("/content/drive/MyDrive/CNN/flowers/rose/118974357_0faa23cce9_n.jpg"))
    plt.axis('off')
    plt.show()

```

predict()

1/1 [=====] - 0s 302ms/step
The given image is a tulip

