Assignment 3

Artificial Intelligence

Assignment Date	31 October 2022
Student Name	S.Karthikeyan
Student Register Number	620619106012
Maximum Marks	2

Build CNN Model for Classification Of Flowers

```
from google.colab import drive
drive.mount('/content/drive')
```

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

1. Downloading and unzipping dataset

```
!unzip '/content/drive/MyDrive/Assignment 3/Flowers-Dataset.zip'
```

```
[ ] !unzip '/content/drive/MyDrive/Assignment 3/Flowers-Dataset.zip'

    Archive:  /content/drive/MyDrive/Assignment 3/Flowers-Dataset.zip
      inflating: flowers/daisy/100080576_f52e8ee070_n.jpg
      inflating: flowers/daisy/10140303196_b88d3d6cec.jpg
      inflating: flowers/daisy/10172379554_b296050f82_n.jpg
      inflating: flowers/daisy/10172567486_2748826a8b.jpg
      inflating: flowers/daisy/10172636503_21bededa75_n.jpg
      inflating: flowers/daisy/102841525_bd6628ae3c.jpg
      inflating: flowers/daisy/10300722094_28fa978807_n.jpg
      inflating: flowers/daisy/1031799732_e7f4008c03.jpg
      inflating: flowers/daisy/10391248763_1d16681106_n.jpg
      inflating: flowers/daisy/10437754174_22ec990b77_m.jpg
      inflating: flowers/daisy/10437770546_8bb6f7bdd3_m.jpg
      inflating: flowers/daisy/10437929963_bc13eebe0c.jpg
      inflating: flowers/daisy/10466290366_cc72e33532.jpg
      inflating: flowers/daisy/10466558316_a7198b87e2.jpg
      inflating: flowers/daisy/10555749515_13a12a026e.jpg
      inflating: flowers/daisy/10555815624_dc211569b0.jpg
      inflating: flowers/daisy/10555826524_423eb8bf71_n.jpg
      inflating: flowers/daisy/10559679065_50d2b16f6d.jpg
      inflating: flowers/daisy/105806915_a9c13e2106_n.jpg
      inflating: flowers/daisy/10712722853_5632165b04.jpg
      inflating: flowers/daisy/107592979_aaa9cdfe78_m.jpg
```

```
inflating: flowers/daisy/10770585085_4742b9dac3_n.jpg
inflating: flowers/daisy/10841136265_af473efc60.jpg
inflating: flowers/daisy/10993710036_2033222c91.jpg
inflating: flowers/daisy/10993818044_4c19b86c82.jpg
inflating: flowers/daisy/10994032453_ac7f8d9e2e.jpg
inflating: flowers/daisy/11023214096_b5b39fab08.jpg
inflating: flowers/daisy/11023272144_fce94401f2_m.jpg
inflating: flowers/daisy/11023277956_8980d53169_m.jpg
inflating: flowers/daisy/11124324295_503f3a0804.jpg
inflating: flowers/daisy/1140299375_3aa7024466.jpg
inflating: flowers/daisy/11439894966_dca877f0cd.jpg
inflating: flowers/daisy/1150395827_6f94a5c6e4_n.jpg
inflating: flowers/daisy/11642632_1e7627a2cc.jpg
inflating: flowers/daisy/11834945233_a53b7a92ac_m.jpg
inflating: flowers/daisy/11870378973_2ec1919f12.jpg
inflating: flowers/daisy/11891885265_ccefec7284_n.jpg
inflating: flowers/daisy/12193032636_b50ae7db35_n.jpg
inflating: flowers/daisy/12348343085_d4c396e5b5_m.jpg
inflating: flowers/daisy/12585131704_0f64b17059_m.jpg
inflating: flowers/daisy/12601254324_3cb62c254a_m.jpg
inflating: flowers/daisy/1265350143_6e2b276ec9.jpg
inflating: flowers/daisy/12701063955_4840594ea6_n.jpg
inflating: flowers/daisy/1285423653_18926dc2c8_n.jpg
inflating: flowers/daisy/1286274236_1d7ac84efb_n.jpg
inflating: flowers/daisy/12891819633_e4c82b51e8.jpg
inflating: flowers/daisy/1299501272_59d9da5510_n.jpg
inflating: flowers/daisy/1306119996_ab8ae14d72_n.jpg
inflating: flowers/daisy/1314069875_da8dc023c6_m.jpg
inflating: flowers/daisy/1342002397_9503c97b49.jpg
inflating: flowers/daisy/134409839_71069a95d1_m.jpg
inflating: flowers/daisy/1344985627_c3115e2d71_n.jpg
inflating: flowers/daisy/13491959645_2cd9df44d6_n.jpg
inflating: flowers/daisy/1354396826_2868631432_m.jpg
inflating: flowers/daisy/1355787476_32e9f2a30b.jpg
inflating: flowers/daisy/13583238844_573df2de8e_m.jpg
inflating: flowers/daisy/1374193928_a52320eafa.jpg
```

2. Image Augmentation

```
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenera
tor
import matplotlib.pyplot as plt
batch_size = 32
img_height = 180
img_width = 180
data_dir = "/content/flowers"
train_datagen = ImageDataGenerator(rescale = 1./255, horizontal_
flip = True, vertical_flip = True, zoom_range = 0.2)
x_train = train_datagen.flow_from_directory('/content/flowers',
target_size=(64,64),
class_mode='categorical',
batch_size=100)
```

```
data_augmentation = Sequential(
[
layers.RandomFlip("vertical",input_shape=(img_height, img_width,
 3)),
layers.RandomRotation(0.1),
layers.RandomZoom(0.1),
]
)
```

```
[ ] import numpy as np
    import tensorflow as tf
    from tensorflow.keras import layers
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.preprocessing.image import ImageDataGenerator
    import matplotlib.pyplot as plt
    batch_size = 32
    img_height = 180
    img_width = 180
    data_dir = "/content/flowers"
```

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

```
[ ] x_train = train_datagen.flow_from_directory('/content/flowers',
    target_size=(64,64),
    class_mode='categorical',
    batch_size=100)
```

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

```
[ ] data_augmentation = Sequential(
    [
    layers.RandomFlip("vertical",input_shape=(img_height, img_width, 3)),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
    ]
    )
```

3. Creating Model

```
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,F
latten,Dense
model = Sequential()
training_ds = tf.keras.utils.image_dataset_from_directory(
data_dir,
validation_split=0.2,
subset="training",
seed=57,
image_size=(img_height, img_width),
batch_size=batch_size)
validation_ds = tf.keras.utils.image_dataset_from_directory(
data_dir,
validation_split=0.2,
subset="validation",
seed=107,
image_size=(img_height, img_width),
batch_size=batch_size)
training_ds.class_names
plt.figure(figsize=(7, 7))
for data, labels in training_ds.take(1):
```

```
for i in range(6):
  ax = plt.subplot(2, 3, i + 1)
  plt.imshow(data[i].numpy().astype("uint8"))
  plt.title(training_ds.class_names[labels[i]])
  plt.axis("off")
```

```
[ ] from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
    model = Sequential()
```

```
[ ] training_ds = tf.keras.utils.image_dataset_from_directory(
    data_dir,
    validation_split=0.2,
    subset="training",
    seed=57,
    image_size=(img_height, img_width),
    batch_size=batch_size)
```

```
Found 4317 files belonging to 5 classes.
Using 3454 files for training.
```

```
[ ] validation_ds = tf.keras.utils.image_dataset_from_directory(
    data_dir,
    validation_split=0.2,
    subset="validation",
    seed=107,
    image_size=(img_height, img_width),
    batch_size=batch_size)
```

```
Found 4317 files belonging to 5 classes.
Using 863 files for validation.
```

```
[ ] training_ds.class_names
```

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

```
[ ] plt.figure(figsize=(7, 7))
    for data, labels in training_ds.take(1):
      for i in range(6):
        ax = plt.subplot(2, 3, i + 1)
        plt.imshow(data[i].numpy().astype("uint8"))
        plt.title(training_ds.class_names[labels[i]])
        plt.axis("off")
```

3 a. Convolution Layer

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

```
[ ] model.add(Convolution2D(32, (3,3), activation = "relu", input_shape = (64,64,3) ))
```

3 b. Maxpooling Layer

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

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

3 c. Flatten

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

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

3 d. Hidden/dense layers

```
model.add(Dense(300, activation = "relu"))
model.add(Dense(150, activation = "relu"))
```

```
[ ] model.add(Dense(300, activation = "relu"))
    model.add(Dense(150, activation = "relu"))
```

3 e. Output layer

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

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

4. Compiling Model

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

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

5. Fit the Model

```
model.fit(x_train, epochs = 15, steps_per_epoch = len(x_train))
```

```
[ ] model.fit(x_train, epochs = 15, steps_per_epoch = len(x_train))

    Epoch 1/15
    44/44 [==============================] - 33s 722ms/step - loss: 1.9328 - accuracy: 0.3294
    Epoch 2/15
    44/44 [==============================] - 30s 676ms/step - loss: 1.1435 - accuracy: 0.5200
    Epoch 3/15
    44/44 [==============================] - 32s 720ms/step - loss: 1.0711 - accuracy: 0.5747
    Epoch 4/15
    44/44 [==============================] - 30s 680ms/step - loss: 1.0149 - accuracy: 0.5969
    Epoch 5/15
    44/44 [==============================] - 30s 671ms/step - loss: 0.9642 - accuracy: 0.6201
    Epoch 6/15
    44/44 [==============================] - 30s 669ms/step - loss: 0.9278 - accuracy: 0.6421
    Epoch 7/15
    44/44 [==============================] - 32s 733ms/step - loss: 0.9032 - accuracy: 0.6493
    Epoch 8/15
    44/44 [==============================] - 30s 674ms/step - loss: 0.8751 - accuracy: 0.6597
    Epoch 9/15
    44/44 [==============================] - 30s 681ms/step - loss: 0.8442 - accuracy: 0.6738
    Epoch 10/15
    44/44 [==============================] - 31s 693ms/step - loss: 0.8211 - accuracy: 0.6850
    Epoch 11/15
    44/44 [==============================] - 32s 730ms/step - loss: 0.8088 - accuracy: 0.6861
    Epoch 12/15
    44/44 [==============================] - 31s 687ms/step - loss: 0.7778 - accuracy: 0.7003
    Epoch 13/15
    44/44 [==============================] - 30s 682ms/step - loss: 0.7652 - accuracy: 0.7086
    Epoch 14/15
    44/44 [==============================] - 30s 682ms/step - loss: 0.7466 - accuracy: 0.7079
    Epoch 15/15
    44/44 [==============================] - 31s 701ms/step - loss: 0.7403 - accuracy: 0.7190
    <keras.callbacks.History at 0x7fe2ea323a10>
```

6. Save the Model

```
model.save("flowers.h1")
```

```
[ ] model.save("flowers.h1")
```

7. Test the Model

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model("/content/flowers.h1")
sunflower_img = image.load_img('/content/flowers/sunflower/10085
66138_6927679c8a.jpg',target_size=(64,64))
x = image.img_to_array(sunflower_img)
x = np.expand_dims(x,axis=0)
predicted_class=model.predict(x)
labels = ['daisy','dandelion','roses','sunflowers','tulips']
labels[np.argmax(predicted_class)]
sunflower_img
```

```
[ ] from tensorflow.keras.models import load_model
    from tensorflow.keras.preprocessing import image
```

```
[ ] model = load_model("/content/flowers.h1")
```

```
[ ] sunflower_img = image.load_img('/content/flowers/sunflower/1008566138_6927679c8a.jpg',target_size=(64,64))
    x = image.img_to_array(sunflower_img)
    x = np.expand_dims(x,axis=0)
    predicted_class=model.predict(x)
```

```
1/1 [==============================] - 0s 43ms/step
```

```
[ ] labels = ['daisy','dandelion','roses','sunflowers','tulips']
    labels[np.argmax(predicted_class)]
```

```
'sunflowers'
```

```
[ ] sunflower_img
```