Assignment -3

Build CNN Model for Classification Of Flowers

ASSIGNMENT DATE	10 October 2022
STUDENT NAME	Hariharan .s
STUDENT ROLLNUMBER	612419104012
MAXIMUM MARKS	2 Marks

Question-1:

- Download the Dataset

SOLUTION:

1)Download the Dataset

```
!unzip "/content/Flowers-Dataset.zip"
```

```
inflating: flowers/daisy/34658035045_7782e95b50_n.jpg
inflating: flowers/daisy/34661399476_9ea7e2fd53_n.jpg
inflating: flowers/daisy/34664107325_701d5c6f08_n.jpg
inflating: flowers/daisy/34665595995_13f76d5b60_n.jpg
inflating: flowers/daisy/34670512115_af22cce24d_n.jpg
inflating: flowers/daisy/34682895116_88ef018e83_n.jpg
inflating: flowers/daisy/3468498624_d082f99e98.jpg
inflating: flowers/daisy/34693373736_9ce6d9e1c3_n.jpg
inflating: flowers/daisy/34695914906_961f92ffcd_n.jpg
inflating: flowers/daisy/34696729796_190b1dfdf1_n.jpg
inflating: flowers/daisy/34696730126_056ffea63c_n.jpg
inflating: flowers/daisy/34696730346_5f0c131e59_n.jpg
inflating: flowers/daisy/34701078235_4a770d14a1_n.jpg
inflating: flowers/daisy/34701198765_54aa641d7a_n.jpg
inflating: flowers/daisy/34718882165_68cdc9def9_n.jpg
inflating: flowers/daisy/34720703615_bdf1335d8b_n.jpg
inflating: flowers/daisy/34727863665_b00ac77266_n.jpg
inflating: flowers/daisy/34729724865_787c98299d_n.jpg
```

Question-2:

- Image Augmentation

SOLUTION:

2)Image Augmentation

```
[ ] from tensorflow.keras.preprocessing.image import ImageDataGenerator
    from sklearn.model_selection import train_test_split

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

[ ] x_train=train_datagen.flow_from_directory(r"/content/flowers",target_size=(64,64),class_mode="categorical",batch_size=32)

    Found 4317 images belonging to 5 classes.

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

[ ] x_test=test_datagen.flow_from_directory(r"/content/flowers",target_size=(64,64),class_mode="categorical",batch_size=32)

    Found 4317 images belonging to 5 classes.

[ ] x_train.class_indices

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

[ ] x_test.class_indices

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

Question-3:

- Create Model & Add Layers

SOLUTION:

3)Create Model & Add Layers

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

```
[ ] #intialize
    model=Sequential()
    model.add(Convolution2D(32,(3,3),strides=(1,1),input_shape=(64,64,3)))
    model.add(Activation("relu"))
    model.add(MaxPooling2D(pool_size=(2,2)))

    model.add(Convolution2D(32,(3,3),strides=(1,1),input_shape=(64,64,3)))
    model.add(Activation("relu"))
    model.add(MaxPooling2D(pool_size=(2,2)))

    model.add(Convolution2D(64,(3,3),strides=(1,1),input_shape=(64,64,3)))
    model.add(Activation("relu"))
    model.add(MaxPooling2D(pool_size=(2,2)))
```

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

```
[ ] model.add(Dense(300))
    model.add(Activation("relu"))
    model.add(Dense(300))
    model.add(Activation("Softmax"))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 32)        896

activation (Activation)      (None, 62, 62, 32)        0

max_pooling2d (MaxPooling2D  (None, 31, 31, 32)        0
)

conv2d_1 (Conv2D)            (None, 29, 29, 32)        9248

activation_1 (Activation)    (None, 29, 29, 32)        0

max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)        0
2D)

conv2d_2 (Conv2D)            (None, 12, 12, 64)        18496
```

Question-4:

- Compile The Model

SOLUTION:

4)Compile The Model

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

```
[ ] len(x_train)

    135
```

Question-5:

- Fit The Model

SOLUTION:

```
model.fit (x_train,epochs=100,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test))
```

```
Epoch 1/10
108/108 [==============================] - 79s 727ms/step - loss: 0.9333 - accuracy: 0.6482 - val_loss: 0.8710 -
val_accuracy: 0.6686
Epoch 2/10
108/108 [==============================] - 78s 724ms/step - loss: 0.8859 - accuracy: 0.6642 - val_loss: 0.9045 -
val_accuracy: 0.6489
Epoch 3/10
108/108 [==============================] - 78s 723ms/step - loss: 0.8349 - accuracy: 0.6769 - val_loss: 0.7972 -
val_accuracy: 0.6918
Epoch 4/10
108/108 [==============================] - 78s 720ms/step - loss: 0.7841 - accuracy: 0.7038 - val_loss: 0.7972 -
val_accuracy: 0.6987
Epoch 5/10
108/108 [==============================] - 78s 720ms/step - loss: 0.7707 - accuracy: 0.7116 - val_loss: 0.7873 -
val_accuracy: 0.7219
Epoch 6/10
108/108 [==============================] - 78s 720ms/step - loss: 0.7255 - accuracy: 0.7171 - val_loss: 0.8026 -
val_accuracy: 0.6952
Epoch 7/10
108/108 [==============================] - 78s 721ms/step - loss: 0.6889 - accuracy: 0.7435 - val_loss: 0.7638 -
val_accuracy: 0.7115
Epoch 8/10
108/108 [==============================] - 78s 720ms/step - loss: 0.6588 - accuracy: 0.7499 - val_loss: 0.7983 -
val_accuracy: 0.7080
Epoch 9/10
108/108 [==============================] - 78s 719ms/step - loss: 0.6502 - accuracy: 0.7522 - val_loss: 0.7979 -
val_accuracy: 0.7173
Epoch 10/10
```

Question-6:

- Save The Model

SOLUTION:

6)Save The Model

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

Question-7:

- Test The Model

SOLUTION: