Assignment -3

Python Programming

Assignment Date	11 October 2022
Student Name	B. Pradeeba
Student Roll Number	953719106031
Maximum Marks	2 Marks

Question-1:

Image Augumentation

Solution:

1. Image Augmentation

```
[ ]  # Import necessary lib.

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

```
[ ]  # Image augmentation on training variable

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

```
[ ]  # Image augmentation on testing variable

     test_datagen = ImageDataGenerator(rescale=1./255)
```

```
[ ]  # Image augmentation on training data

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

```
[ ]  # Image augmentation on testing data

     xtest = test_datagen.flow_from_directory('/content/flowers',
                                              target_size=(64,64),
                                              class_mode='categorical',
                                              batch_size=100)
```

Question-2:

Create Model

Solution:

2.Create the model

```
[ ]  model = Sequential() # Initializing sequential model
```

QUESTION -3

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

Solution:

3.Add Layers

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

```
[ ]  # Importing requested library.

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

```
[ ]  # Build a CNN block

     model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # convolution layer
     model.add(MaxPooling2D(pool_size=(2, 2))) # Max pooling layer
     model.add(Flatten()) # Flatten layer
     model.add(Dense(300,activation='relu')) # Hidden layer 1
     model.add(Dense(150,activation='relu')) # Hidden layer 2
     model.add(Dense(4,activation='softmax')) # Output layer
```

QUESTION-4

Compile the model

Solution:

4.Compile The Model

```
[ ]  # Compiling the model

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

QUESTION-5
Fit the model
Solution:

5.Fit The Model

```
[ ]  # Train model

     model.fit_generator(xtrain,
                         steps_per_epoch=len(xtrain),
                         epochs=10,
                         validation_data=xtest,
                         validation_steps=len(xtest))
```

QUESTION-6
Save the model
Solution:

6.Save The Model

```
[ ]  # Save model

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

QUESTION -7

Test the model

Solution:

7.Test the model

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

```
[ ]  # Testing 1

     img = image.load_img('/flowers/dandelion/33907694863_f7c0f23ef3_n.jpg',target_size=(64,64)) # Reading image
     x = image.img_to_array(img) # Converting image into array
     x = np.expand_dims(x,axis=0) # expanding Dimensions
     pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
     op = [daisy', 'dandelion', 'rose', 'sunflower', 'tulip'] # Creating list
     op[pred] # List indexing with output
```

```
[ ]  # Testing 2

     img = image.load_img('/flowers/daisy/1150395827_6f94a5c6e4_n.jpg',target_size=(64,64)) # Reading image
     x = image.img_to_array(img) # Converting image into array
     x = np.expand_dims(x,axis=0) # expanding Dimensions
     pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
     op = [daisy', 'dandelion', 'rose', 'sunflower', 'tulip'] # Creating list
     op[pred] # List indexing with output
```