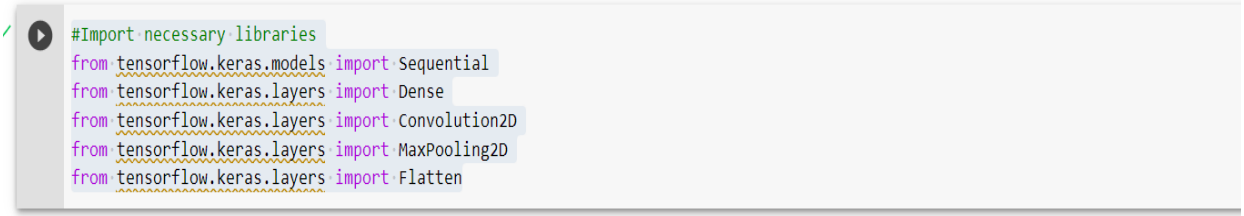
Assignment -3

Convolution Neural Network

Assignment Date	6 October 2022
Student Name	Sineka G
Student Roll Number	9517201906047
Maximum Marks	2 Marks

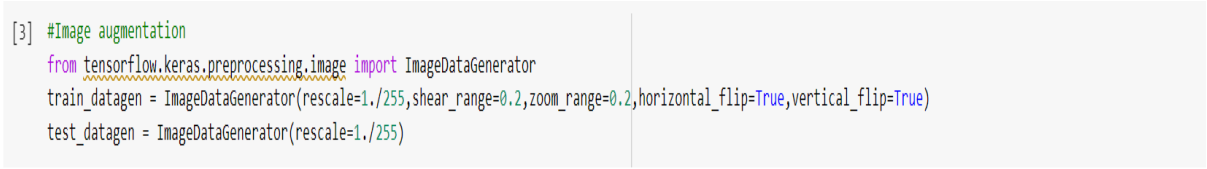
#Import necessary libraries

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



#Image augmentation

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen =
ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_fl
ip=True,vertical_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)
```



#Dataset

```
x_train=train_datagen.flow_from_directory(r"E:\Flowers\Training",target_size=(
128,128),batch_size=32,class_mode="categorical")
x_test=test_datagen.flow_from_directory(r"E:\Flowers\Testing",target_size=(128
,128),batch_size=32,class_mode="categorical")
```

x_train.class_indices

```
[4] x_train = train_datagen.flow_from_directory(r"/content/drive/MyDrive/flowers/Training",target_size=(128,128),batch_size=32,class_mode="categorical")
    x_test = test_datagen.flow_from_directory(r"/content/drive/MyDrive/flowers/Testing",target_size=(128,128),batch_size=32,class_mode="categorical")
    x_train.class_indices
```

```
Found 3023 images belonging to 5 classes.
Found 1325 images belonging to 5 classes.
{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}
```

#Add layers

```
model = Sequential()
```

#Convolution layer

```
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
```

#Maxpooling layer

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

#Flatten layer

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

#Hidden layer

```
model.add(Dense(units=300,kernel_initializer="random_uniform",activation="relu
"))
model.add(Dense(units=200,kernel_initializer="random_uniform",activation="relu
"))
model.add(Dense(units=5,kernel_initializer="random_uniform",activation="softma
x"))
model.summary()
```

```
[5] model = Sequential()
    #Add layers
    #Convolution layer
    model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
    #Maxpooling layer
    model.add(MaxPooling2D(pool_size=(2,2)))
    #flatten layer
    model.add(Flatten())
    #hidden layer
    model.add(Dense(units=300,kernel_initializer="random_uniform",activation="relu"))
    model.add(Dense(units=200,kernel_initializer="random_uniform",activation="relu"))
    model.add(Dense(units=5,kernel_initializer="random_uniform",activation="softmax"))
    model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      896

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 flatten (Flatten)           (None, 127008)            0

 dense (Dense)               (None, 300)               38102700

 dense_1 (Dense)             (None, 200)               60200

 dense_2 (Dense)             (None, 5)                 1005

=================================================================
```

#Compile the model

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

#Fit the model

```
model.fit_generator(x_train,steps_per_epoch=75,epochs=15,validation_data=x_test,validation_steps=80)
```

```
[6] #compile the model
    model.compile(loss="categorical_crossentropy",optimizer="adam",metrics=["accuracy"])
    #Fit the model
    model.fit_generator(x_train,steps_per_epoch=75,epochs=15,validation_data=x_test,validation_steps=80)

    /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`
      after removing the cwd from sys.path.
    Epoch 1/15
    75/75 [==============================] - ETA: 0s - loss: 1.4611 - accuracy: 0.3729WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your data
    75/75 [==============================] - 986s 13s/step - loss: 1.4611 - accuracy: 0.3729 - val_loss: 1.2855 - val_accuracy: 0.4611
    Epoch 2/15
    75/75 [==============================] - 111s 1s/step - loss: 1.1692 - accuracy: 0.5233
    Epoch 3/15
    75/75 [==============================] - 50s 670ms/step - loss: 1.0733 - accuracy: 0.5749
    Epoch 4/15
    75/75 [==============================] - 36s 472ms/step - loss: 1.0288 - accuracy: 0.5976
    Epoch 5/15
    75/75 [==============================] - 35s 468ms/step - loss: 0.9513 - accuracy: 0.6229
    Epoch 6/15
    75/75 [==============================] - 34s 446ms/step - loss: 0.8990 - accuracy: 0.6555
    Epoch 7/15
    75/75 [==============================] - 33s 443ms/step - loss: 0.8472 - accuracy: 0.6643
    Epoch 8/15
    75/75 [==============================] - 35s 462ms/step - loss: 0.8096 - accuracy: 0.6971
    Epoch 9/15
    75/75 [==============================] - 33s 444ms/step - loss: 0.8063 - accuracy: 0.6928
    Epoch 10/15
    75/75 [==============================] - 33s 442ms/step - loss: 0.8184 - accuracy: 0.6878
    Epoch 11/15
    75/75 [==============================] - 33s 446ms/step - loss: 0.7711 - accuracy: 0.7104
    Epoch 12/15
    75/75 [==============================] - 34s 452ms/step - loss: 0.7128 - accuracy: 0.7268
    Epoch 13/15
```

#Save the model

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

```
[7] #Save the model
    model.save("flower.h5")
```

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
model = load_model("Flower.h5")
```

```
[8] from tensorflow.keras.models import load_model
    from tensorflow.keras.preprocessing import image
    import numpy as np
    model = load_model("flower.h5")
```

#Test the model

```
img = image.load_img(r"C:\Users\hp\Downloads\rose.jpg",target_size=(128,128))
img
type(img)
```

```
[24] #Testing with the image
     img = image.load_img(r"/content/daisy.jpg",target_size=(128,128))
     img
     type(img)
```

```
PIL.Image.Image
```

```
x = image.img_to_array(img)
x
x.shape
x = np.expand_dims(x,axis=0)
x.shape
```

```
[20] x = image.img_to_array(img)
     x
     x.shape
     x = np.expand_dims(x,axis=0)
     x.shape
```

```
(1, 128, 128, 3)
```

```
pred_prob = model.predict(x)
pred_prob
```

```
[21] pred_prob = model.predict(x)
     pred_prob
```

```
array([[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 0.0000000e+00,
        1.2715527e-29]], dtype=float32)
```

```
class_name = ["daisy","dandelion","rose","sunfower","tulip"]
```

```
[22] class_name = ["daisy","dandelion","rose","sunfower","tulip"]
```

```
pred_id = pred_prob.argmax(axis=1)[0]
pred_id
print("Predicted flower is",str(class_name[pred_id]))
```

```
[23] pred_id = pred_prob.argmax(axis=1)[0]
     pred_id
     print("Predicted flower is",str(class_name[pred_id]))
```

```
Predicted flower is daisy
```