|  |
| --- |
|  |
|  |  | drive.mount('/content/drive') |
|  |  | Mounted at /content/drive |
|  |  | Image Augmentation |
|  |  | # Importing Library |
|  |  | from tensorflow.keras.preprocessing.image import ImageDataGenerator |
|  |  | # expanding training and testing variable |
|  |  | train\_d=ImageDataGenerator(rescale=1./255,zoom\_range=0.2,horizontal\_flip=True) |
|  |  | test\_d=ImageDataGenerator(rescale=1./255) |
|  |  | #Data augmentation on testing data |
|  |  | vtrain = train\_d.flow\_from\_directory('/content/drive/MyDrive/flowers/Testing',target\_size=(76,76),class\_mode='categorical',batch\_size=200) |
|  |  | Found 4334 images belonging to 5 classes. |
|  |  | #Data augmentation on training data |
|  |  | vtest = test\_d.flow\_from\_directory('/content/drive/MyDrive/flowers/Training',target\_size=(76,76),class\_mode='categorical',batch\_size=200) |
|  |  | Found 4372 images belonging to 5 classes. |
|  |  | Creating CNN Model |
|  |  | #Importing Libraries |
|  |  | from tensorflow.keras.models import Sequential |
|  |  | from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense |
|  |  | #Building a CNN block |
|  |  | model = Sequential() |
|  |  | model.add(Convolution2D(32,(3,3),activation='relu',input\_shape=(76,76,3))) |
|  |  | model.add(MaxPooling2D(pool\_size=(2, 2))) |
|  |  | model.add(Flatten()) |
|  |  | model.add(Dense(500,activation='relu')) |
|  |  | model.add(Dense(250,activation='relu')) |
|  |  | model.add(Dense(5,activation='softmax')) |
|  |  | #Compiling the model |
|  |  |  |
|  |  | model.compile(optimizer='adam',loss='categorical\_crossentropy',metrics=['accuracy']) |
|  |  | #Fittting the model |
|  |  |  |
|  |  | model.fit\_generator(vtrain,steps\_per\_epoch=len(vtrain),epochs=15,validation\_data=vtest,validation\_steps=len(vtest)) |
|  |  | /usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:3: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators. |
|  |  | This is separate from the ipykernel package so we can avoid doing imports until |
|  |  | Epoch 1/15 |
|  |  | 22/22 [==============================] - 1419s 66s/step - loss: 2.4632 - accuracy: 0.2995 - val\_loss: 1.2851 - val\_accuracy: 0.4376 |
|  |  | Epoch 2/15 |
|  |  | 22/22 [==============================] - 64s 3s/step - loss: 1.2076 - accuracy: 0.5074 - val\_loss: 1.1575 - val\_accuracy: 0.5320 |
|  |  | Epoch 3/15 |
|  |  | 22/22 [==============================] - 62s 3s/step - loss: 1.0800 - accuracy: 0.5743 - val\_loss: 1.0376 - val\_accuracy: 0.5958 |
|  |  | Epoch 4/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.9855 - accuracy: 0.6214 - val\_loss: 0.9492 - val\_accuracy: 0.6414 |
|  |  | Epoch 5/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.8937 - accuracy: 0.6622 - val\_loss: 0.9133 - val\_accuracy: 0.6530 |
|  |  | Epoch 6/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.8337 - accuracy: 0.6751 - val\_loss: 0.7866 - val\_accuracy: 0.7091 |
|  |  | Epoch 7/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.7875 - accuracy: 0.7037 - val\_loss: 0.7907 - val\_accuracy: 0.7100 |
|  |  | Epoch 8/15 |
|  |  | 22/22 [==============================] - 65s 3s/step - loss: 0.7410 - accuracy: 0.7220 - val\_loss: 0.6903 - val\_accuracy: 0.7434 |
|  |  | Epoch 9/15 |
|  |  | 22/22 [==============================] - 65s 3s/step - loss: 0.7011 - accuracy: 0.7323 - val\_loss: 0.6207 - val\_accuracy: 0.7699 |
|  |  | Epoch 10/15 |
|  |  | 22/22 [==============================] - 66s 3s/step - loss: 0.6562 - accuracy: 0.7575 - val\_loss: 0.6067 - val\_accuracy: 0.7793 |
|  |  | Epoch 11/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.6345 - accuracy: 0.7637 - val\_loss: 0.7020 - val\_accuracy: 0.7381 |
|  |  | Epoch 12/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.6324 - accuracy: 0.7649 - val\_loss: 0.5490 - val\_accuracy: 0.8008 |
|  |  | Epoch 13/15 |
|  |  | 22/22 [==============================] - 63s 3s/step - loss: 0.6061 - accuracy: 0.7695 - val\_loss: 0.5225 - val\_accuracy: 0.8118 |
|  |  | Epoch 14/15 |
|  |  | 22/22 [==============================] - 65s 3s/step - loss: 0.5382 - accuracy: 0.8032 - val\_loss: 0.4787 - val\_accuracy: 0.8255 |
|  |  | Epoch 15/15 |
|  |  | 22/22 [==============================] - 66s 3s/step - loss: 0.5271 - accuracy: 0.8050 - val\_loss: 0.5410 - val\_accuracy: 0.8001 |
|  |  | <keras.callbacks.History at 0x7f9e94a95e90> |
|  |  | # save model |
|  |  | model.save('flowers.h5') |
|  |  | Testing model |
|  |  | from tensorflow.keras.preprocessing import image |
|  |  | import numpy as np |
|  |  | # Testing 1.1(daisy) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/daisy/10993818044\_4c19b86c82.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'daisy' |
|  |  | # Testing 1.2(daisy) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/daisy/525780443\_bba812c26a\_m.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'daisy' |
|  |  | # Testing 2.1(dandelion) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/dandelion/1195255751\_d58b3d3076.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'sunflower' |
|  |  | # Testing 2.2(dandelion) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/dandelion/1297972485\_33266a18d9.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'daisy' |
|  |  | # Testing 3.1(rose) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/rose/7456887736\_54e4ebac03\_n.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'rose' |
|  |  | # Testing 3.2(rose) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/rose/33411423082\_8150d9254e\_n.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'rose' |
|  |  | # Testing 4.1(sunflower) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/sunflower/7012364067\_5ffc7654c9\_m.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'sunflower' |
|  |  | # Testing 4.2(sunflower) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/sunflower/2720698862\_486d3ec079\_m.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'sunflower' |
|  |  | # Testing 5.1(tulip) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/tulip/8892851067\_79242a7362\_n.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'tulip' |
|  |  | # Testing 5.2(tulip) |
|  |  |  |
|  |  | img = image.load\_img('/content/drive/MyDrive/flowers/Testing/tulip/5546723510\_39a5a10d3a\_n.jpg',target\_size=(76,76)) |
|  |  | x = image.img\_to\_array(img) |
|  |  | x = np.expand\_dims(x,axis=0) |
|  |  | prediction = np.argmax(model.predict(x)) |
|  |  | op = ['daisy','dandelion','rose','sunflower','tulip'] |
|  |  | op[prediction] |
|  |  | 'tulip' |