19ITR066

Pushpamala R

Importation of libraries

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
%matplotlib inline
import os
import PIL
import cv2
import tensorflow as tf
from·tensorflow·import·keras
from·tensorflow.keras·import·layers,models
```

Loading DatasetPath

```
import pathlib
data_dir=pathlib.Path("/content/drive/MyDrive/MachineLearning/flowers1")
data_dir
```

```
PosixPath('/content/drive/MyDrive/MachineLearning/flowers1')
```

Fetching no of images in dataset

```
len(list(data_dir.glob('*/*.jpg')))
```

```
4317
```

Loading images under its image names as dictionary

```
flower_images={
    'roses' : list(data_dir.glob('roses/*.jpg')),
    'tulips': list(data_dir.glob('tulips/*.jpg')),
    'dandelin':list(data_dir.glob('dandelin/*.jpg')),
    'sunflowers':list(data_dir.glob('sunflowers/*.jpg')),
    'daisy':list(data_dir.glob('daisy/*.jpg'))
}
```

Assigning Labels for the names

```
flower_labels={
    'roses':0,
    'tulips':1,
    'dandelin':2,
    'sunflowers':3,
    'daisy':4
}
```

Creating images list and its output label list

```
x,y=[],[]
for flower_name,images in flower_images.items():
    for image in images:
        img = cv2.imread(str(image))
        resized_img=cv2.resize(img,(180,180))
        x.append(resized_img)
        y.append(flower_labels[flower_name])
```

```
x = np.array(x)
y = np.array(y)
```

Splitting Training and Testing Data

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)
```

Scaling - Preprocessing

```
X_train_scaled = x_train / 255
X_test_scaled = x_test / 255
```

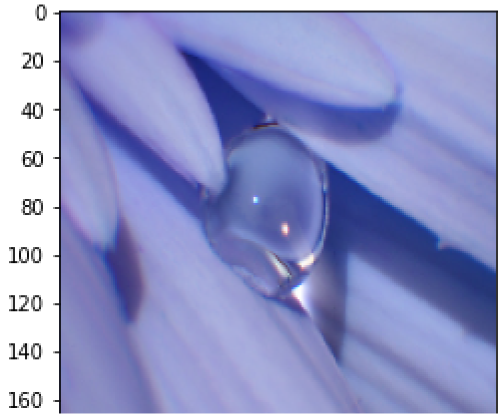
Image Augmentation

```
data_augmentation = keras.Sequential(
  [
    layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"
                                                 ),
    layers.experimental.preprocessing.RandomRotation(0.2),
    layers.experimental.preprocessing.RandomZoom(0.1)
  ]
)
```

Original image and its augmented image

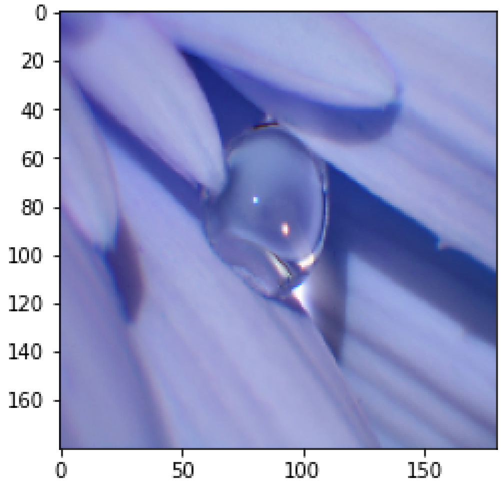
```
plt.imshow(x[0])
```

```
<matplotlib.image.AxesImage at 0x7f617aacdb50>
```

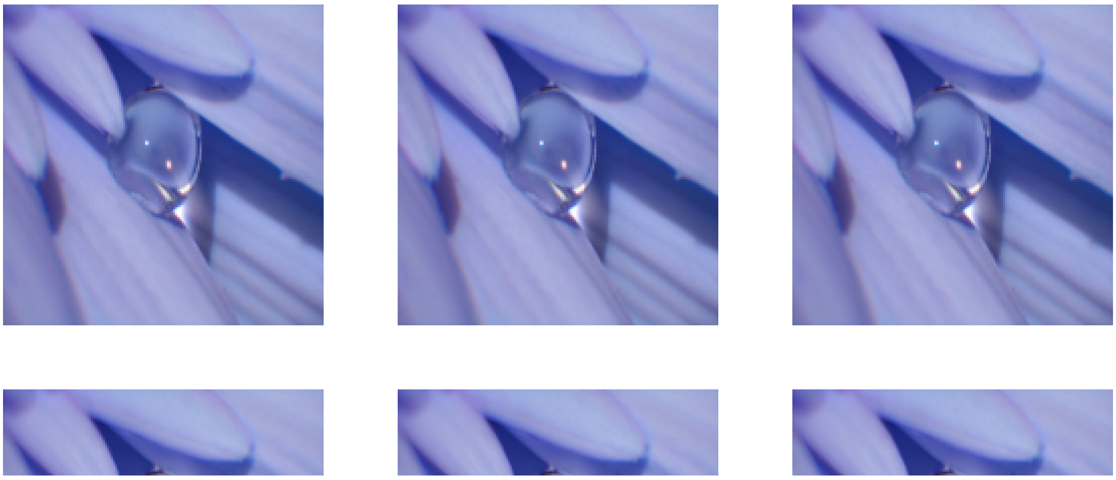


```
plt.imshow(data_augmentation(x)[0].numpy().astype("uint8"))
```

```
<matplotlib.image.AxesImage at 0x7f617afbd690>
```



```
plt.figure(figsize=(10, 10))
for i in range(9):
  augmented_image = data_augmentation(x)
  ax = plt.subplot(3, 3, i + 1)
  plt.imshow(augmented_image[0].numpy().astype("uint8"))
  plt.axis("off")
```



```
num_classes = 5

model = keras.Sequential([
  data_augmentation,
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Dropout(0.2),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])

model.fit(X_train_scaled, y_train, epochs=5)
```

```
Epoch 1/5
18/18 [==============================] - 21s 1s/step - loss: 0.0877 - accuracy: 0.966
Epoch 2/5
18/18 [==============================] - 20s 1s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 3/5
18/18 [==============================] - 21s 1s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 4/5
18/18 [==============================] - 28s 2s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 5/5
18/18 [==============================] - 25s 1s/step - loss: 0.0000e+00 - accuracy: 1
<keras.callbacks.History at 0x7f617ad5d250>
```

Model Creation

```
num_classes = 5

model = keras.Sequential([
  layers.Conv2D(16, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(32, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Conv2D(64, 3, padding='same', activation='relu'),
  layers.MaxPooling2D(),
  layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])
```

```
Epoch 1/5
18/18 [==============================] - 18s 945ms/step - loss: 0.0947 - accuracy: 0
Epoch 2/5
18/18 [==============================] - 17s 957ms/step - loss: 0.0000e+00 - accuracy
Epoch 3/5
18/18 [==============================] - 17s 953ms/step - loss: 0.0000e+00 - accuracy
Epoch 4/5
18/18 [==============================] - 18s 993ms/step - loss: 0.0000e+00 - accuracy
Epoch 5/5
18/18 [==============================] - 17s 945ms/step - loss: 0.0000e+00 - accuracy
<keras.callbacks.History at 0x7f617b1ab650>
```

Compilation of model

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

Train a model

```
model.fit(X_train_scaled, y_train, epochs=5)
```

```
Epoch 1/5
18/18 [==============================] - 22s 1s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 2/5
18/18 [==============================] - 20s 1s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 3/5
18/18 [==============================] - 20s 1s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 4/5
18/18 [==============================] - 21s 1s/step - loss: 0.0000e+00 - accuracy: 1
Epoch 5/5
18/18 [==============================] - 20s 1s/step - loss: 0.0000e+00 - accuracy: 1
<keras.callbacks.History at 0x7f617e055b10>
```

Saving model

Double-click (or enter) to edit

```
import joblib as jbl


filename = "Completed_model.pkl"
jbl.dump(model, filename)
```

```
['Completed_model.pkl']
```

Load model

```
model=jbl.load("Completed_model.pkl")
```

Model Evaluation

Double-click (or enter) to edit

```
model.evaluate(X_test_scaled,y_test)
```

```
6/6 [==============================] - 2s 283ms/step - loss: 0.0000e+00 - accuracy: 1
[0.0, 1.0]
```