Image Preprocessing

DATE	17 November 2022
TEAM ID	PNT2022TMID14614
PROJECT NAME	AI-powered Nutrition Analyzer for Fitness Enthusiasts

In this milestone, we will be improving the image data that suppresses unwilling distortions or enhances some image features important for further processing, although performing some geometric transformations of images like rotation, scaling, translation, etc.

1)Import The Generalization Library
2)Configure Generalization Class
3)Apply Image Data-generator Functionality To Train-set And Test-set

Import The ImageDataGenerator Library

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset.

The Keras deep learning neural network library provides the capability to fit models using image data augmentation via the ImageDataGenerator class.
Let us import the ImageDataGenerator class from Keras

Configure ImageDataGenerator Class

ImageDataGenerator class is instantiated and the configuration for the types of data augmentation

There are five main types of data augmentation techniques for image data; specifically:

Image shifts via the width_shift_range and height_shift_range arguments.
The image flips via the horizontal_flip and vertical_flip arguments.
Image rotations via the rotation_range argument
Image brightness via the brightness_range argument.
Image zoom via the zoom_range argument.

An instance of the ImageDataGenerator class can be constructed for train and test.

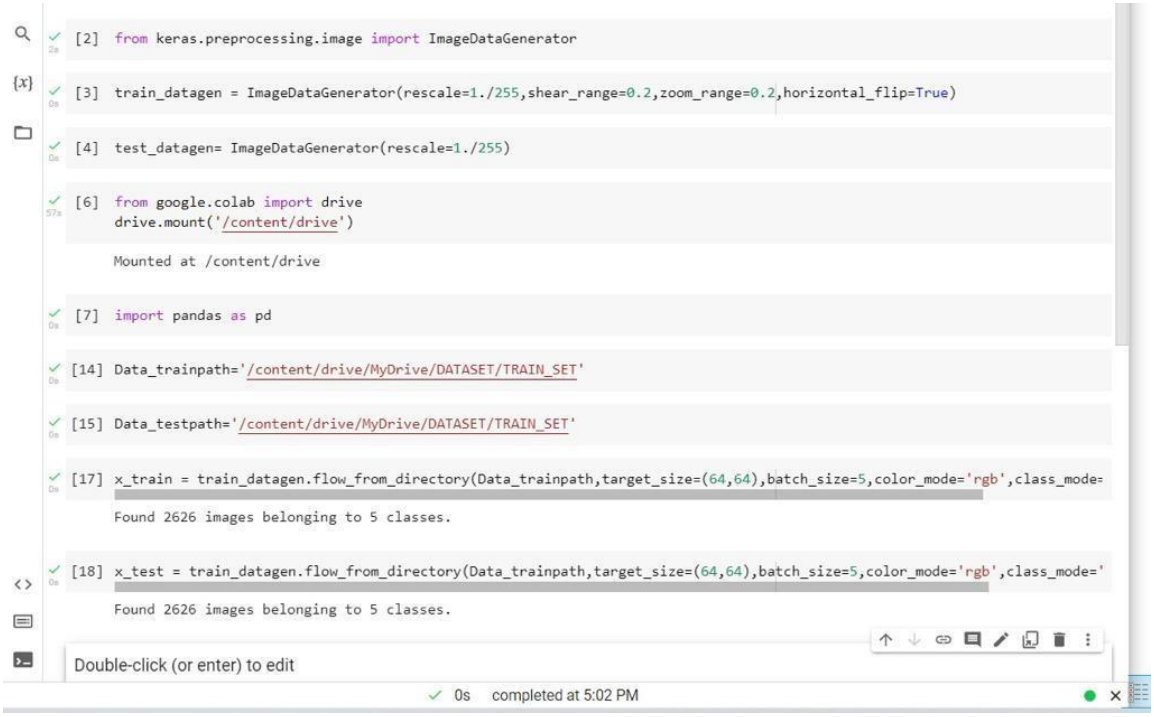
Apply Image DataGenerator Functionality To Trainset And Testset

Let us apply ImageDataGenerator functionality to Trainset and Testset by using the following code
For Training set using flow_from_directory function.

This function will return batches of images from the subdirectories'apples', 'banana', 'orange', 'pineapple', 'watermelon' together with labels 0 to 4{'apples': 0, 'banana': 1, 'orange': 2, 'pineapple': 3, 'watermelon': 4}

Arguments:

- directory: Directory where the data is located. If labels are "inferred", it should contain subdirectories, each containing images for a class. Otherwise, the directory structure is ignored.
- batch_size: Size of the batches of data. Default: 32.
- target_size: Size to resize images after they are read from disk.
- class_mode:
  - ‘int': means that the labels are encoded as integers (e.g. for sparse_categorical_crossentropy loss).
  - 'categorical' means that the labels are encoded as a categorical vector (e.g. for categorical_crossentropy loss).
  - 'binary' means that the labels (there can be only 2) are encoded as float32 scalars with values 0 or 1 (e.g. for binary_crossentropy).
  - None (no labels).



```
[2] from keras.preprocessing.image import ImageDataGenerator

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

[4] test_datagen= ImageDataGenerator(rescale=1./255)

[6] from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive

[7] import pandas as pd

[14] Data_trainpath='/content/drive/MyDrive/DATASET/TRAIN_SET'

[15] Data_testpath='/content/drive/MyDrive/DATASET/TRAIN_SET'

[17] x_train = train_datagen.flow_from_directory(Data_trainpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode=

    Found 2626 images belonging to 5 classes.

[18] x_test = train_datagen.flow_from_directory(Data_trainpath,target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='

    Found 2626 images belonging to 5 classes.

Double-click (or enter) to edit

0s  completed at 5:02 PM
```