Image Preprocessing

DATE	06 Nov 2022
TEAM ID	PNT2022TMID25338
PROJECT NAME	AI-powered Nutrition Analyzer for Fitness Enthusiasts

Image Preprocessing

In this point of reference, we are going be making strides the picture information that smothers unwilling twists or upgrades a few picture highlights imperative for assist handling, in spite of the fact that performing a few geometric changes of pictures like turn, scaling, interpretation, etc.

Loading and pre-processing the data:

Deep learning models view data as the precious metal. A large training set of photos increases the likelihood that your image classification model will perform effectively. Additionally, the data's shape changes depending on the architecture and framework we employ.

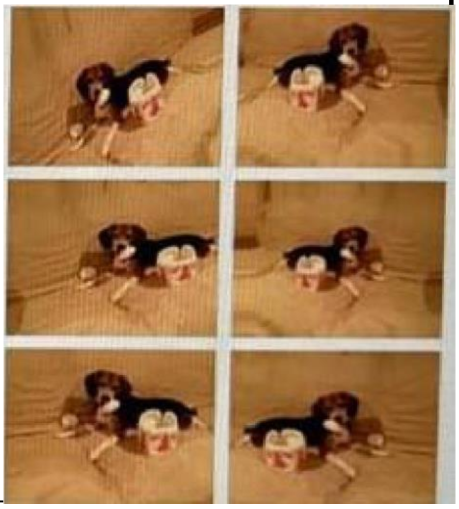
The crucial data pre-processing phase is as a result (the eternally important step in any project). I strongly advise reading through "Basics of Image Processing Using Python." To perform data augmentation, we use the Kera Image Data Generator class. In other words, we are processing the data we've gathered using some form of parameters. The term "augment" refers to the act of making something (in this example, data) "bigger" or "increased." The Kera Image Data Generator class actually functions by

Approving a collection of training-related photos. Applying a number of arbitrary adjustments to each picture in the batch (including random rotation, resizing, shearing, etc.).
the new, randomly altered batch is substituted for the initial batch. Using this batch of randomly modified data, train the CNN (i.e., the original data itself is not used for training).

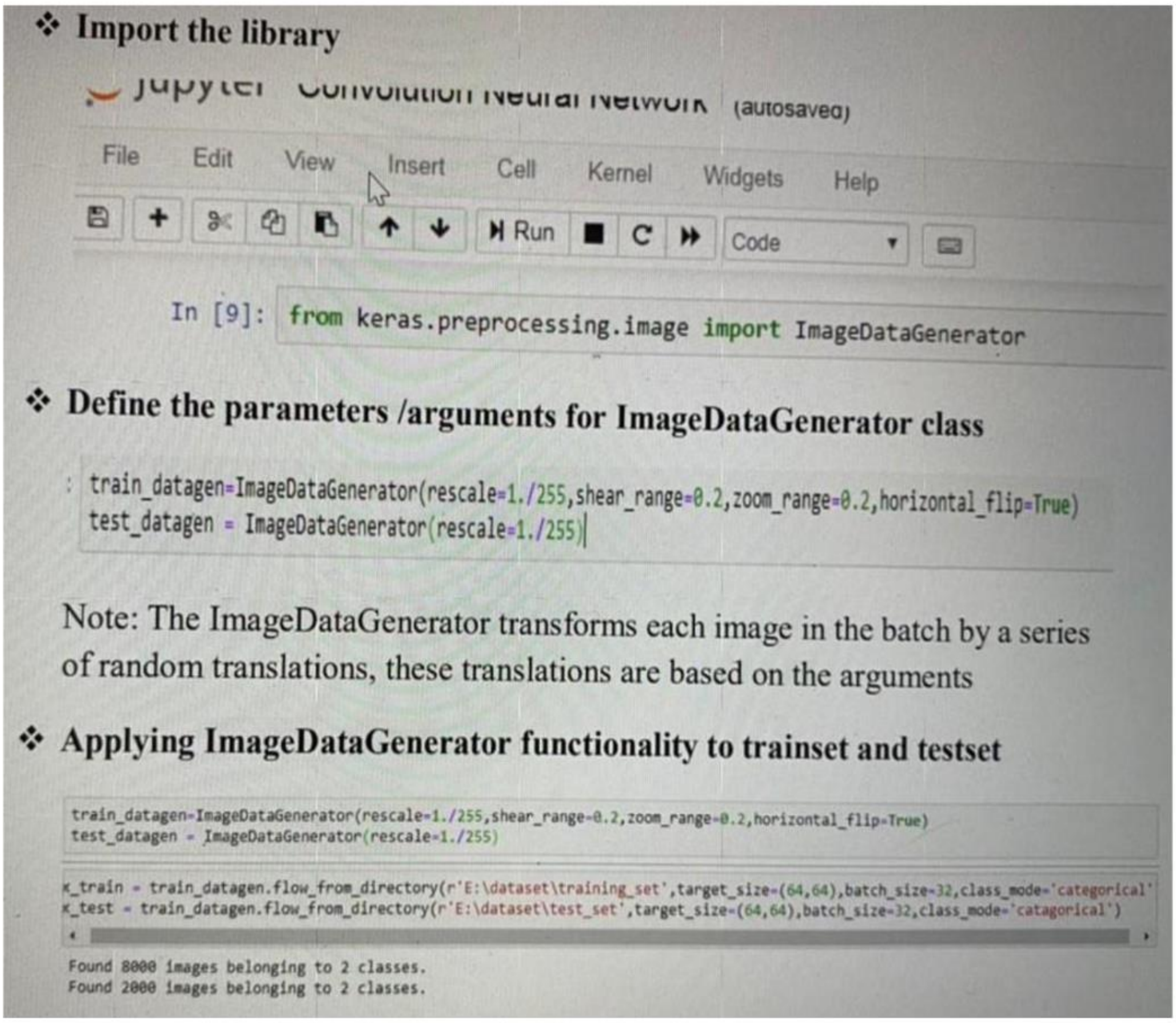
Note: The Image Data Generator only returns the newly converted data after randomly transforming the original data.

Input Image

Augrn,ent ed Images



II Keras



❖ **Import the library**

jupyter Convolution Neural Network (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run Code

```
In [9]: from keras.preprocessing.image import ImageDataGenerator
```

❖ **Define the parameters /arguments for ImageDataGenerator class**

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

Note: The ImageDataGenerator transforms each image in the batch by a series of random translations, these translations are based on the arguments

❖ **Applying ImageDataGenerator functionality to trainset and testset**

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

```
x_train = train_datagen.flow_from_directory(r'E:\dataset\training_set',target_size=(64,64),batch_size=32,class_mode='categorical'
x_test = train_datagen.flow_from_directory(r'E:\dataset\test_set',target_size=(64,64),batch_size=32,class_mode='catagorical')
```

```
Found 8000 images belonging to 2 classes.
Found 2000 images belonging to 2 classes.
```