Image Pre-processing Phase
Apply Image DataGenerator Functionality To Trainset And Testset

Date	08 November 2022
Team ID	PNT2022TMID12370
Project Name	AI-Powered Nutrition Analyzer For Fitness Enthusiasts

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).

```
### Loading our data and performing data agumentation

#performing data agumentation to train data
x_train = train_datagen.flow_from_directory(
    r'C:\Users\DELL\Desktop\Desk Files\Nutrition Analysis Using Image Classification\DataSet\TRAIN_SET',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
#performing data agumentation to test data
x_test = test_datagen.flow_from_directory(
    r'C:\Users\DELL\Desktop\Desk Files\Nutrition Analysis Using Image Classification\DataSet\TEST_SET',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')

Found 2626 images belonging to 5 classes.
Found 1055 images belonging to 5 classes.
```

We notice that 2626 images are belonging to 5 classes for training and 1055 images belong to 5 classes for testing purposes.

```
print(x_train.class_indices)#checking the number of classes

 {'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}

print(x_test.class_indices)#checking the number of classes

 {'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}

from collections import Counter as c
c(x_train .labels)

 Counter({0: 606, 1: 445, 2: 479, 3: 621, 4: 475})
```

Here we are checking the number of classes in train and test data and counting the number of images in each class of train set data by using the counter function.

Screenshot:

Apply Image DataGenerator Functionality To Trainset And Testset

[16]:
```
#Performing data augmentation to train data
x_train = train_datagen.flow_from_directory(r'D:\Nutrition Analysis Using Image Classification\Dataset\TRAIN_SET',
                                            target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')

#Performing data augmentation to test data
x_test = test_datagen.flow_from_directory(r'D:\Nutrition Analysis Using Image Classification\Dataset\TEST_SET',
                                          target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
```

```
Found 8129 images belonging to 5 classes.
Found 1060 images belonging to 5 classes.
```

[17]:
```
#Checking the number of classes
print(x_train.class_indices)
```

```
{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}
```

[18]:
```
#Checking the number of classes
print(x_test.class_indices)
```

```
{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}
```

[19]:
```
from collections import Counter as c
c(x_train .labels)
```

[19]:
```
Counter({0: 2441, 1: 1843, 2: 1998, 3: 820, 4: 1027})
```