

Emerging methods for early detection of forest fire

MODEL BUILDING

INITIALIZING THE MODEL

Team ID	PNT2022TMID34027
Project Name	Emerging methods for early detection of forest fire

INITIALILIZING THE MODEL: keras

has 2 ways to define a neural network:

- Sequential
- Function API

The Sequential class is used to define linear initializations of network layers which then, collectively, constitute a model. In our example below, we will use the Sequential constructor to create a model, which will then have layers added to it using the add () method.

Now, will initialize our model.

▾ Importing Keras libraries

```
import keras
```

▾ Importing ImageDataGenerator from Keras

```
from keras.preprocessing.image import ImageDataGenerator
```

▾ Importing Keras libraries

```
✓ [1] import keras
```

▾ Importing ImageDataGenerator from Keras

```
✓ [13] from matplotlib import pyplot as plt  
from keras.preprocessing.image import ImageDataGenerator
```

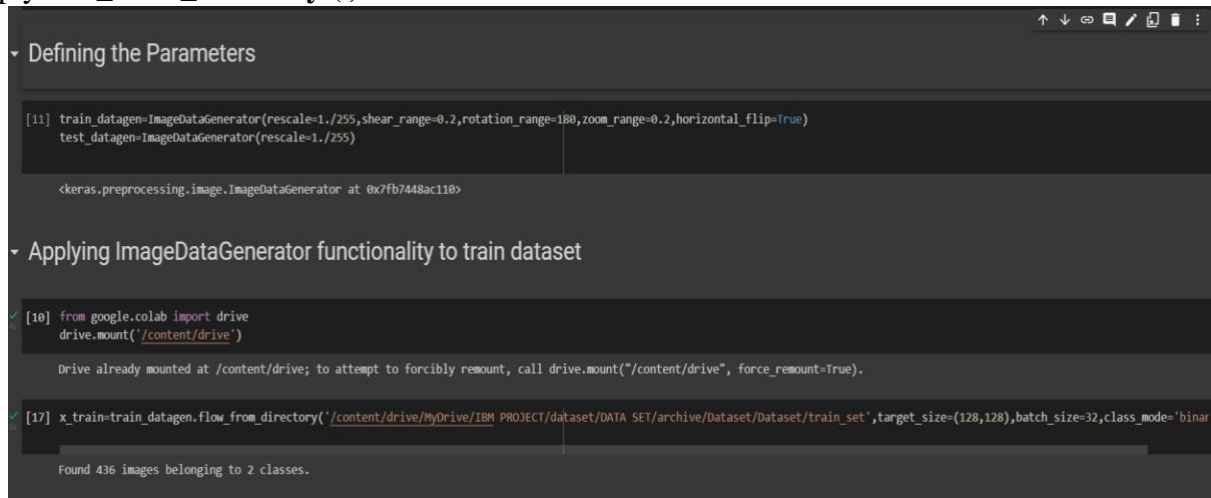
▾ Defining the Parameters

```
▶ train_datagen=ImageDataGenerator(rescale=1./255, shear_range=0.2, rotation_range=180, zoom_range=0.2, horizontal_flip=True)  
test_datagen=ImageDataGenerator(rescale=1./255)
```

```
□ <keras.preprocessing.image.ImageDataGenerator at 0x7fb7448ac110>
```

APPLYING ImageDataGenerator to train dataset:

ply `flow_from_directory ()` method for Train folder.



The screenshot shows a Jupyter Notebook with two sections. The first section, 'Defining the Parameters', contains a code cell [11] that initializes `train_datagen` and `test_datagen` as `ImageDataGenerator` objects with parameters: `rescale=1./255`, `shear_range=0.2`, `rotation_range=180`, `zoom_range=0.2`, and `horizontal_flip=True`. The second section, 'Applying ImageDataGenerator functionality to train dataset', contains two code cells. Cell [10] mounts the Google Drive at `/content/drive`. Cell [17] uses `train_datagen.flow_from_directory` to process the training data from `/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/train_set` with `target_size=(128,128)`, `batch_size=32`, and `class_mode='binary'`. The output of cell [17] states: 'Found 436 images belonging to 2 classes.'

```
defining the Parameters

[11] train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotation_range=180,zoom_range=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)

<keras.preprocessing.image.ImageDataGenerator at 0x7fb7448ac110>

Applying ImageDataGenerator functionality to train dataset

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

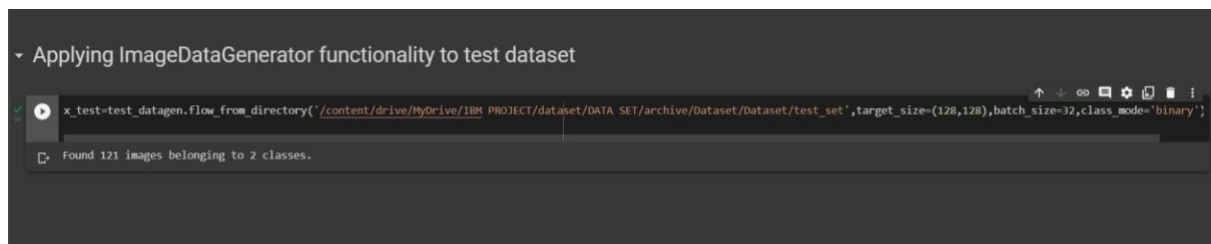
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[17] x_train=train_datagen.flow_from_directory('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/train_set',target_size=(128,128),batch_size=32,class_mode='binary')

Found 436 images belonging to 2 classes.
```

APPLYING ImageDataGenerator to test dataset:

Applying the `flow_from_directory ()` method for test folder.



The screenshot shows a Jupyter Notebook with a section titled 'Applying ImageDataGenerator functionality to test dataset'. It contains a code cell [18] that uses `test_datagen.flow_from_directory` to process the test data from `/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/test_set` with `target_size=(128,128)`, `batch_size=32`, and `class_mode='binary'`. The output of the cell states: 'Found 121 images belonging to 2 classes.'

```
Applying ImageDataGenerator functionality to test dataset

[18] x_test=test_datagen.flow_from_directory('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/test_set',target_size=(128,128),batch_size=32,class_mode='binary')

Found 121 images belonging to 2 classes.
```

IMPORTING MODEL BUILDING LIBRARIES:

Importing Model Building Libraries

```
#to define the linear Initialisation import sequential
from keras.models import Sequential
#to add layers import Dense
from keras.layers import Dense
#to create Convolutional kernel import convolution2D
from keras.layers import Convolution2D
#import Maxpooling layer
from keras.layers import MaxPooling2D
#import flatten layer
from keras.layers import Flatten
import warnings
warnings.filterwarnings('ignore')
```

INITIALIZING THE MODEL:

▼ Initializing the model

```
model=Sequential()
```