

EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

MODEL BUILDING

ADDING CNN LAYERS

Team ID	PNT2022TMID12754
Project Name	Emerging Methods for Early Detection of Forest Fires

ADDING CNN LAYERS:

We will be adding three layers for CNN

- Convolution layer
- Pooling layer
- Flattening layer

Adding Convolution Layer:

The convolution layer is the first and core layer of CNN. It is one of the building blocks of a CNN and is used for extracting important features from the image.

The convolution operation involves the input image with a feature detector/filter to get a feature map. The key role of feature detectors is to extract features from images. A set of feature maps is called a feature layer.

In the convolution2D function we provided arguments containing 32(3,3). This means applying 32 filters of 3x3 matrix filters, input_shape is the input image shape in RGB. where 64x64 is the size and 3 represents the channels. RGB color image.

Activation features:

These are features that help determine if a node should be activated. These functions introduce non-linearity into the network.

Adding Pooling Layer :

Max Pooling selects the largest elements from the area of the feature map covered by the filter. So the output after the max pooling layer will be a feature map containing the most salient features of the previous feature map.

A pooling layer is added after the convolution layer. You can add a max pooling layer using the MaxPooling2D class. It takes a pool size as a parameter. The effective size of the pooling matrix is (2,2). Returns a pooled feature map. (Note: You can add any number of convolution, pooling, and dropout layers.)

In the above code, pool_size refers to pooling filter or kernel size.

Task 3: Adding Flatten Layer

A pooled feature map is transformed from a pooling layer to a one-dimensional matrix or map, where each pixel in one column is just flat. A flattening layer transforms a multidimensional matrix into a one-dimensional layer.

IMPORT LIBRARIES:

11/7/22, 12:35 AM

Untitled8.ipynb - Colaboratory

▼ Importing Keras libraries

```
import keras
```

▼ Importing ImageDataGenerator from Keras

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

IMPORT ImageDataGenerator FROM KERAS:

▼ 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:

`flow_from_directory ()` method for Train folder.

```
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.

```
Applying ImageDataGenerator functionality to test dataset

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:

11/8/22, 1:16 AM

Main code - Colaboratory

▾ 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()
```

ADDING CNN LAYERS:

▼ Adding CNN Layers

```
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
#add maxpooling layers
model.add(MaxPooling2D(pool_size=(2,2)))
#add faltten layer
model.add(Flatten())
```