EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

MODEL BUILDING

ADDING CNN LAYERS

Date	29 October 2022
Team ID	PNT2022TMID03761
Project Name	Emerging Methods for Early Detection of Forest Fires

- Deep Learning – which has emerged as an effective tool for analyzing big data – uses complex algorithms and artificial neural networks to train machines/computers so that they can learn from experience, classify and recognize data/images just like a human brain does.
- Within Deep Learning, a Convolutional Neural Network or CNN is a type of artificial neural network, which is widely used for image/object recognition and classification.
- Deep Learning thus recognizes objects in an image by using a CNN. CNNs are playing a major role in diverse tasks/functions like image processing problems, computer vision tasks like localization and segmentation, video analysis, to recognize obstacles in self-driving cars, as well as speech recognition in natural language processing.
- As CNNs are playing a significant role in these fast-growing and emerging areas, they are very popular in Deep Learning.

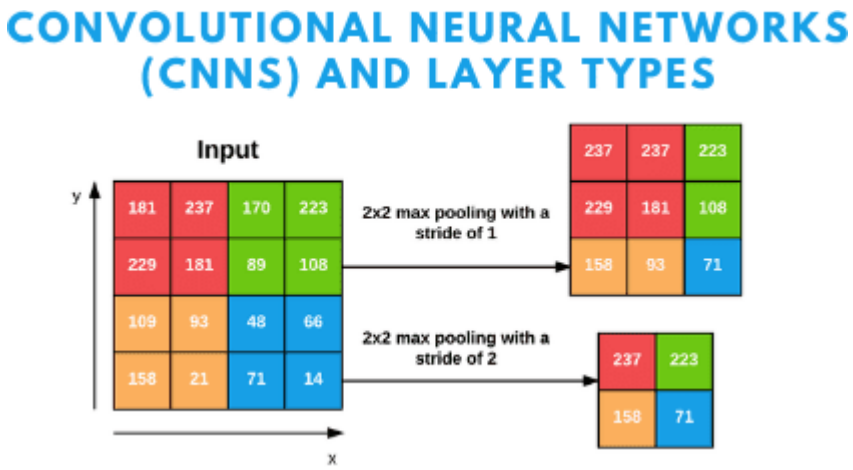
Adding CNN layers:

We will be adding three layers for CNN

- Convolution layer
- Pooling layer
- Flattening layer

Task 1: Adding Convolutional Layer

- A convolutional layer is **the main building block of a CNN**. It contains a set of filters (or kernels), parameters of which are to be learned throughout the training.



- The size of the filters is usually smaller than the actual image. Each filter convolves with the image and creates an activation map.
    - A convolutional layer is the main building block of a CNN. It contains a set of filters (or kernels), parameters of which are to be learned throughout the training. The size of the filters is usually smaller than the actual image. Each filter convolves with the image and creates an activation map.

- A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data.
- The convolutional 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.
- In the Convolution operation, the input image will be convolved with the feature detector/filters to get a feature map. The important role of the feature detector is to extract the features from the image. The group of feature maps is called a feature layer.

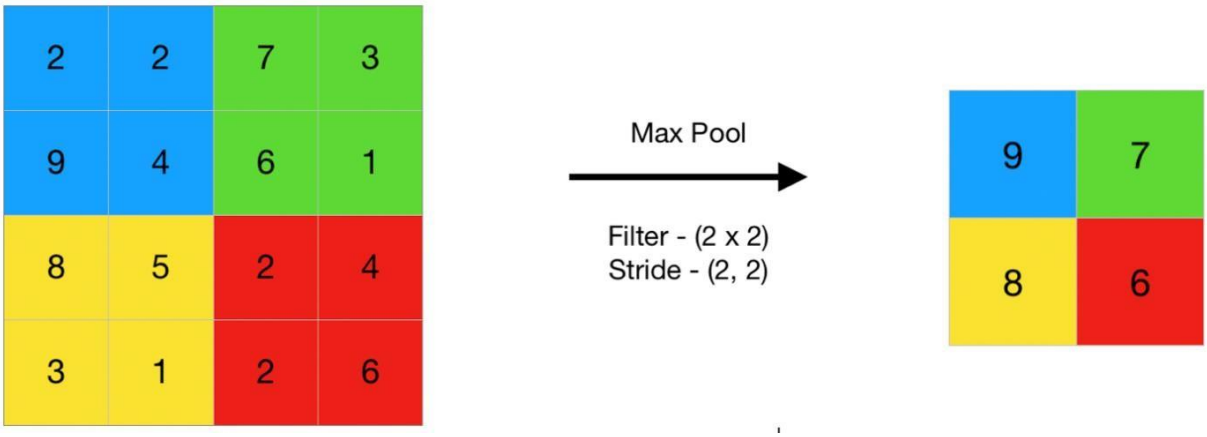
```
#add convolutional layer
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
```

In the convolution2D function, we gave arguments that include 32,(3,3), that refers to we are applying 32 filters of 3x3 matrix filter, and input shape is the input image shape with RGB, here 64x64 is the size and 3 represent the channel, RGB colour images.

Activation Function: These are the functions that help us to decide if we need to activate the node or not. These functions introduce non-linearity in the networks.

Task 2: Adding Pooling Layer:

- The purpose of the pooling layers is to reduce the dimensions of the hidden layer by combining the outputs of neuron clusters at the previous layer into a single neuron in the next layer.



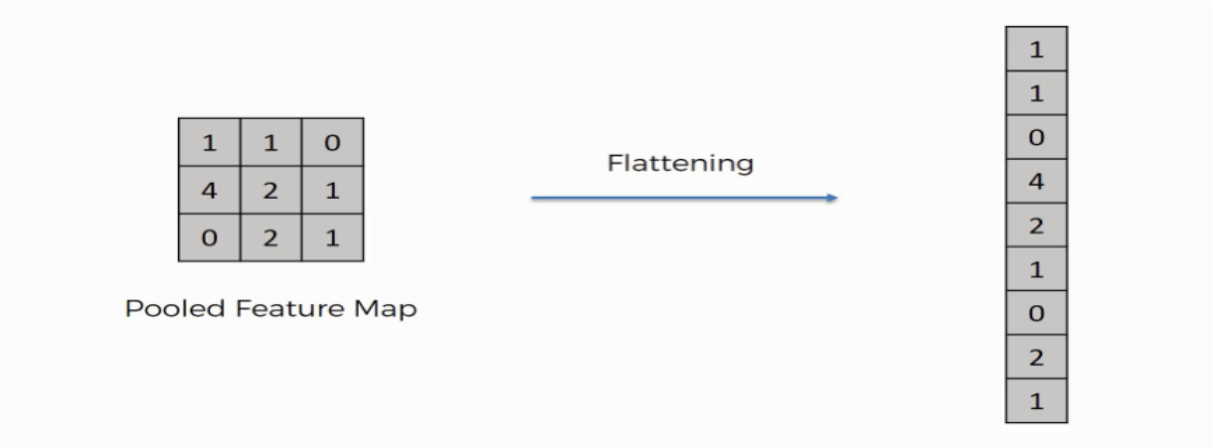
- Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map.
- Pooling mainly helps in extracting sharp and smooth features. It is also done to reduce variance and computations. Max-pooling helps in extracting low-level features like edges, points, etc. While Avg-pooling goes for smooth features.
- Max Pooling selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map.
    - After the convolution layer, a pooling layer is added. Max pooling layer can be added using MaxPooling2D class. It takes the pool size as a parameter. Efficient size of the pooling matrix is (2,2). It returns the pooled feature maps. (Note: Any number of convolution layers, pooling and dropout layers can be added)

```
#add maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
```

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

Adding Flatten Layer:

- Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector. The flattened matrix is fed as input to the fully connected layer to classify the image.



- Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector. The flattened matrix is fed as input to the fully connected layer to classify the image.
- Use of flatten layer- Flatten layer is used to make the multidimensional input one-dimensional, commonly used in the transition from the convolution layer to the full connected layer. Based on whether TensorSpace Model load a pre-trained model before initialization, configure Layer in different ways
- A tensor flatten operation is a common operation inside convolutional neural networks. This is because convolutional layer

outputs that are passed to fully connected layers must be flatted out before the fully connected layer will accept the input.

Now the pooled feature map from the pooling layer will be converted into one single dimension matrix or map, where each pixel in one single column, nothing but flattening. The flattening layer converts the multi-dimension matrix to one single dimension layer.

```
In [ ]: #add flatten layer
        model.add(Flatten())
```