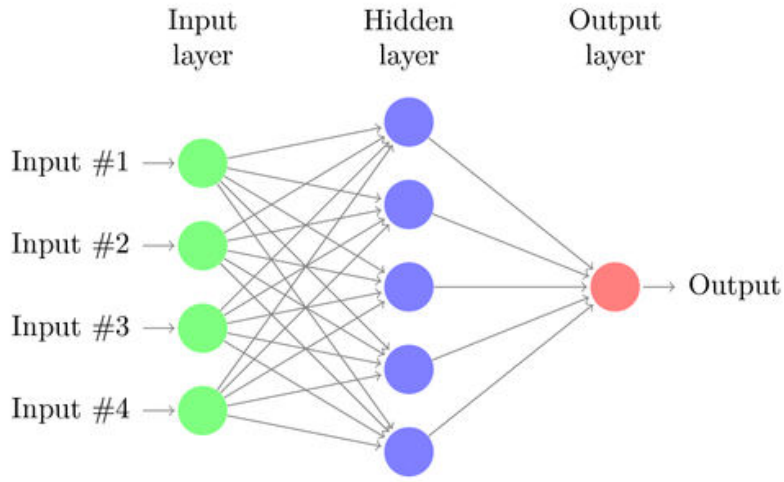
CNN (Convolutional Neural Network)

The Problem with Traditional Neural Networks



A standard multilayer perceptron (traditional neural network).

2

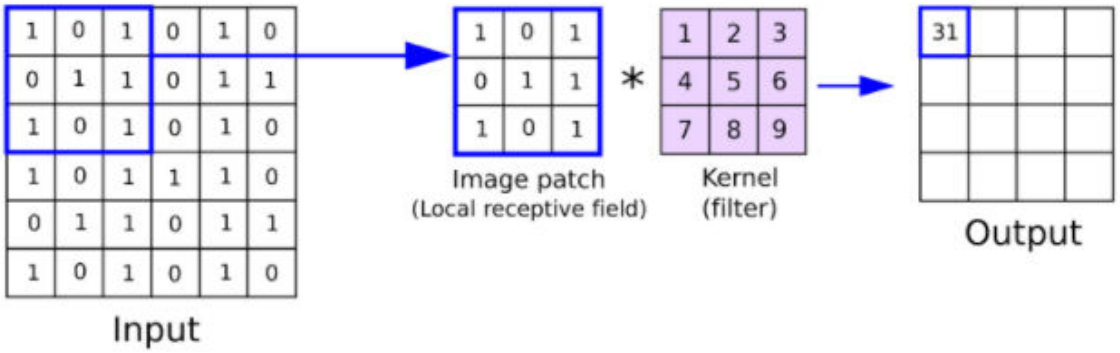
There are several drawbacks of MLP's, especially when it comes to image processing. MLPs use one perceptron for each input (e.g. pixel in an image, multiplied by 3 in RGB case). The amount of weights rapidly becomes unmanageable for large images. For a 24 x 224 pixel image with 3 color channels there are around 150,000 weights that must be trained! As a result, difficulties arise while training and overfitting can occur.

Another common problem is that MLPs react differently to an input (images).

Convolution in CNN:

The CNN is a special type of neural network model designed to work on images data that can be one-dimensional, two-dimensional, and sometimes three-dimensional. Their application ranges from image and video recognition, image classification, medical image analysis, computer vision and natural language processing.

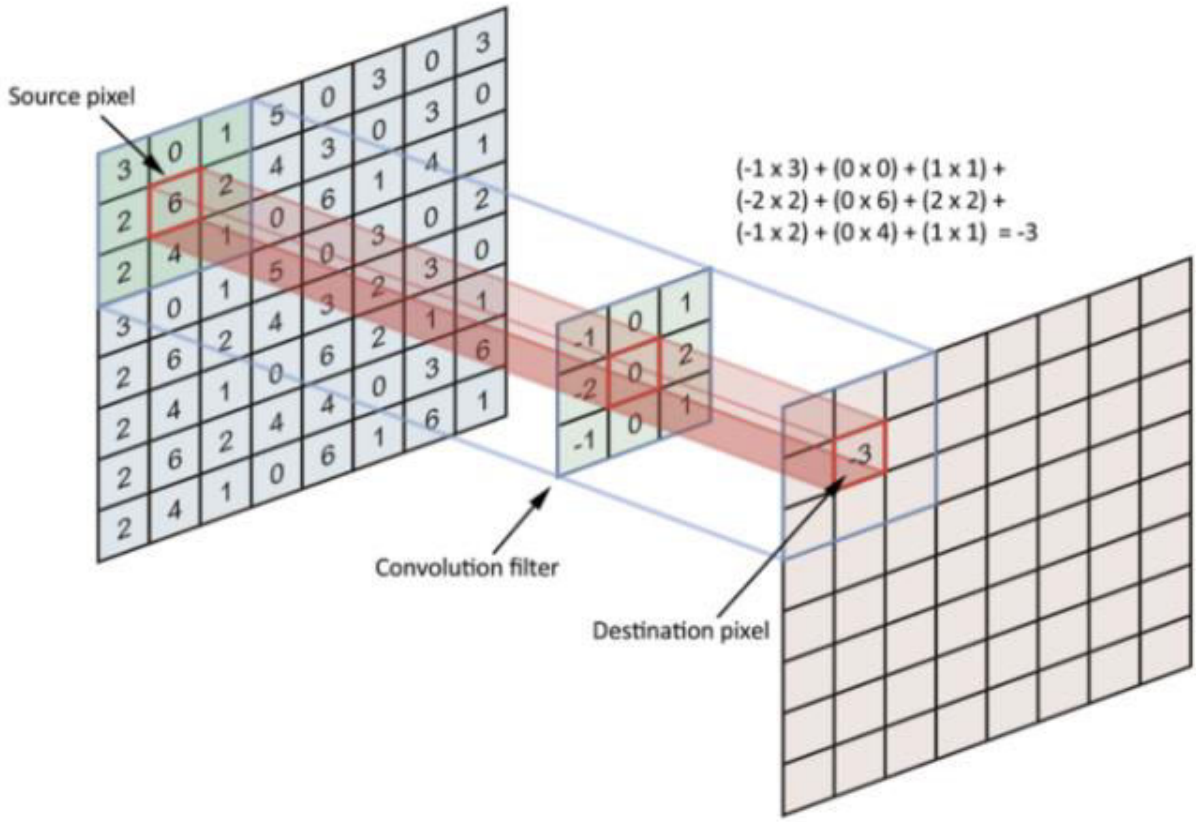
In the context of CNN, convolution is a linear operation involving the multiplication of a set of weights with the input images represented by metrics similar to traditional neural networks. Here an array of weights is called a filter or kernel.



Explain 1D, 2D, 3D Convolution operation with figures.

Strides define the motion of the filter; if you set stride=1, which is the default value, the kernel takes one step at a time.

Usually, the filter size is smaller than the input data, and the type multiplication applied between filter and filter sized sample of input data is the dot product. A dot product is an element-wise multiplication between filter weights and filter sized sample of input data, summed up in a single value.



After the filters have passed over the image, a feature map is generated for each filter. These are then taken through an activation function, which decides whether a certain feature is present at a given location in the image. We can then do a lot of things, such as adding more filtering layers and creating more feature maps, which become more and more abstract as we create a deeper CNN.

There are three hyperparameters which affect the volume size of the output that need to be set before the training of the neural network begins. These include:

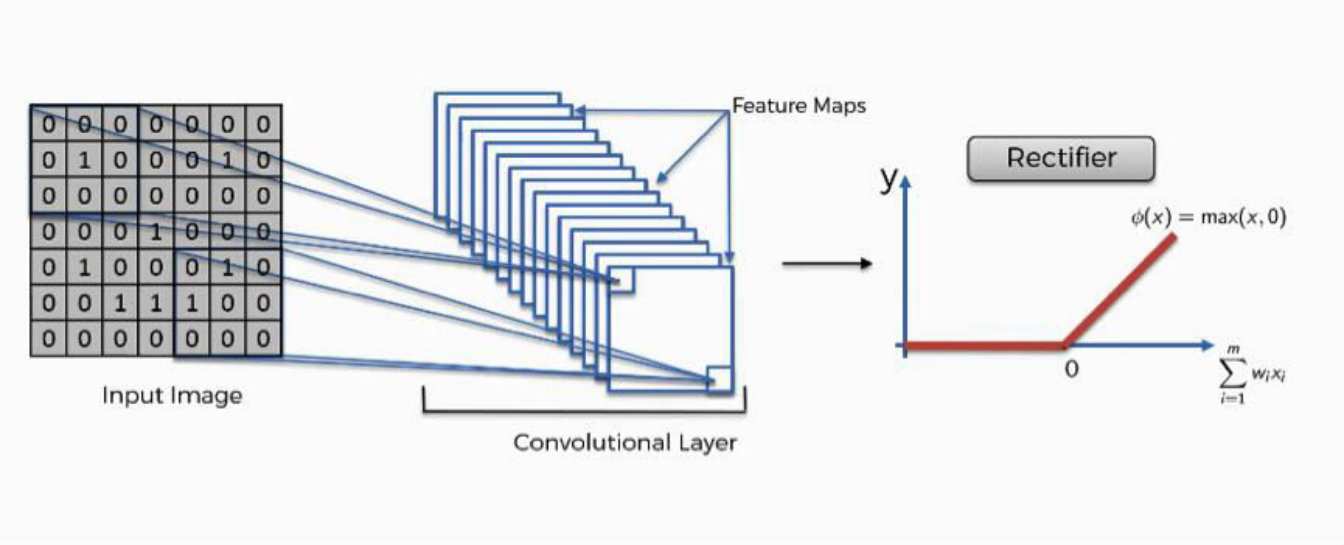
1. The **number of filters** affects the depth of the output. For example, three distinct filters would yield three different feature maps, creating a depth of three.

2. **Stride** is the distance, or number of pixels, that the kernel moves over the input matrix. While stride values of two or greater is rare, a larger stride yields a smaller output.

3. **Zero-padding** is usually used when the filters do not fit the input image. This sets all elements that fall outside of the input matrix to zero, producing a larger or equally sized output.

After each convolution operation, a CNN applies a Rectified Linear Unit (ReLU) transformation to the feature map, introducing nonlinearity to the model.

The Rectified Linear Unit, or ReLU, is not a separate component of the convolutional neural networks' process.



The purpose of applying the rectifier function is to increase the non-linearity in our images. The reason we want to do that is that images are naturally non-linear. The **rectified linear activation function** or **ReLU** for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation function for many

types of neural networks because a model that uses it is easier to train and often achieves better performance.

The input image
This black and white image is the original input image.

Feature detector
By putting the image through the convolution process, the result is what you see in the following image.

As you see, the entire image is now composed of pixels that vary from white to black with many shades of gray in between.

Rectification
What the rectifier function does to an image like this is remove all the black elements from it, keeping only those carrying a positive value (the grey and white colors).

The essential difference between the non-rectified version of the image and the rectified one is the progression of colors. If you look closely at the first one, you will find parts where a white streak is followed by a grey one and then a black one.

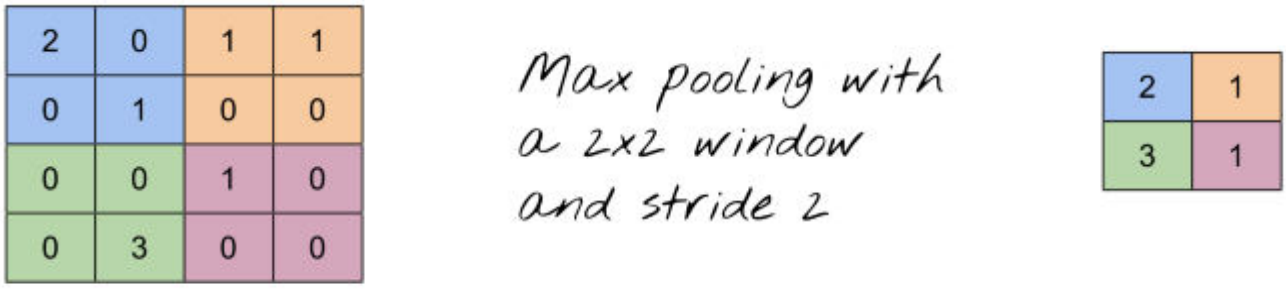
After we rectify the image, you will find the colors changing more abruptly. The gradual change is no longer there. That indicates that the linearity has been disposed of.

Pooling

Note that pooling is a separate step from convolution. Pooling is used to reduce the image size of width and height. Note that the depth is determined by the number of channels. As the name suggests, all it does is it picks the maximum value in a certain size of the window. Although it's usually applied spatially to reduce the x, y dimensions of an image.

Max-Pooling

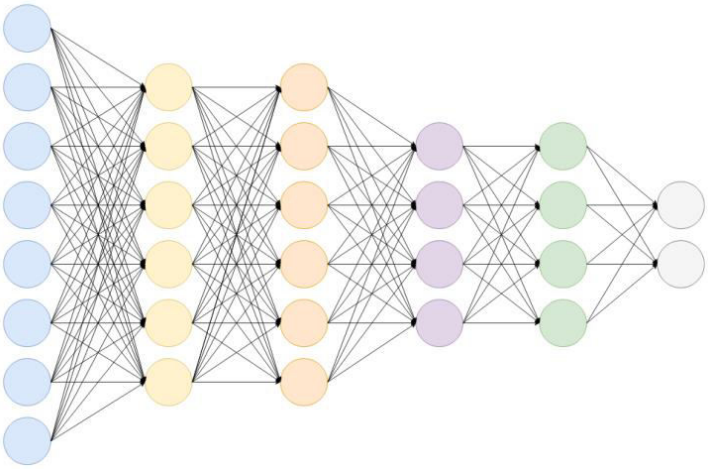
Max pooling is used to reduce the image size by mapping the size of a given window into a single result by taking the maximum value of the elements in the window.



Average-Pooling

It's same as max-pooling except that it averages the windows instead of picking the maximum value.

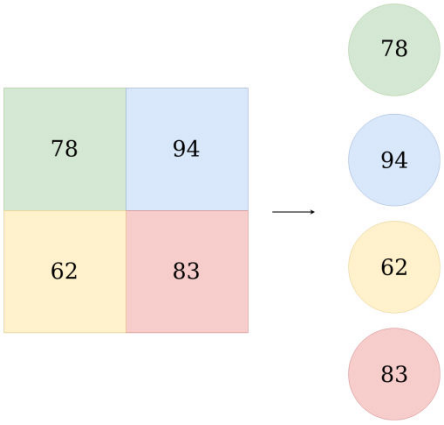
Fully connected layers



Fully Connected Layer is simply, *feed forward neural networks.* Fully Connected Layers form the last few layers in the network. The **input** to the fully connected layer is the output from the *final* Pooling or Convolutional Layer, which is ***flattened*** and then fed into the fully connected layer.

Flattened *means?*

The output from the final Pooling and Convolutional Layer is a 3-dimensional matrix, to flatten that is to unroll all its values into a vector.



This Flattened vector is then connected to a few fully connected layers which are same as Artificial Neural Networks and perform the same mathematical operations. For each layer of the Artificial Neural Network, the following calculation takes place.

$$g(Wx + b)$$

where,

x — *is the input vector*

W — *Is the weight matrix*

b — *Is the bias vector*

g — *Is the activation function, which is usually* ***ReLU****.*

This calculation is repeated for each layer.

After passing through the fully connected layers, the final layer uses the ***softmax activation function*** *(instead of ReLU)* which is used to get probabilities of the input being in a particular class (*classification*).

https://towardsdatascience.com/convolutional-neural-network-17fb77e76c05#:~:text=Fully%20Connected%20Layer,-

Fig%204.&text=Fully%20Connected%20Layer%20is%20simply,into%20the%20fully%20connected%20layer.

LeNet

https://www.analyticsvidhya.com/blog/2021/03/the-architecture-of-lenet-5/