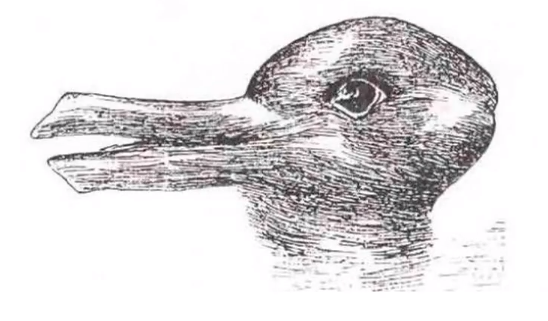
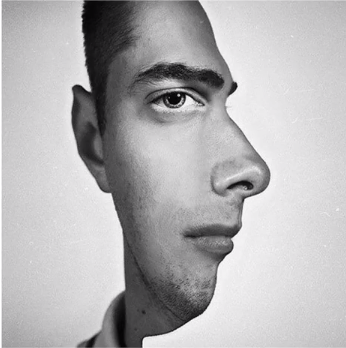
**Convolutional Neural Networks**

What are Convolutional Neural Networks?

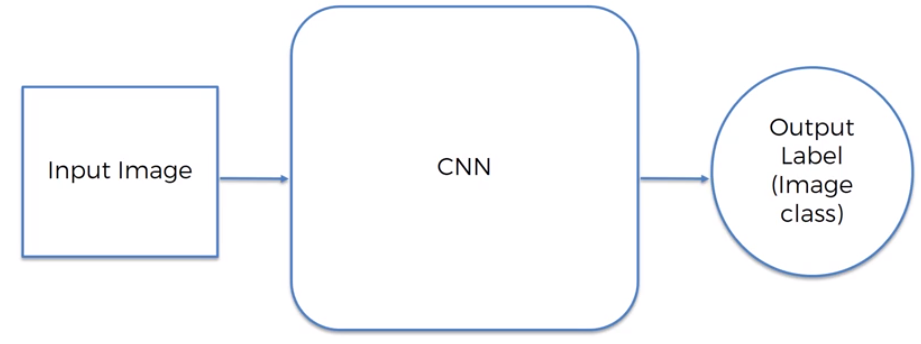
Brain looks for features while analyzing the images. It processes certain features on an image and then classifies it.

Image Illusions –

How convolutional NN work?

It has an input image, it goes through CNN and we have a output label. The image is classified into a output label.

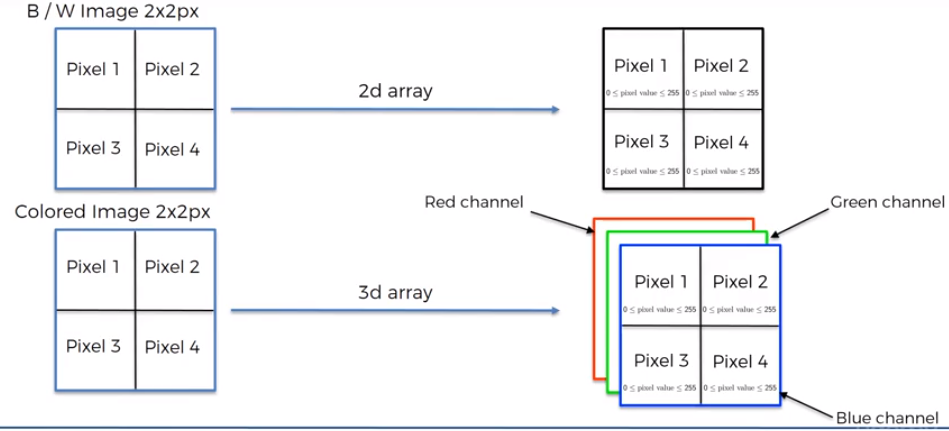


NN can be trained to recognize expressions of a person – Happy, frowning.

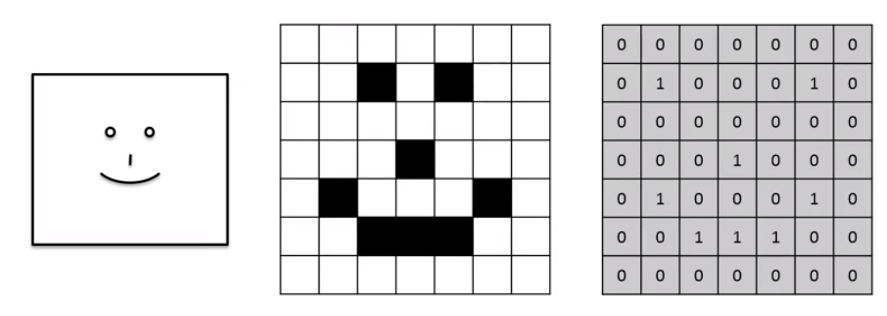
Consider two images – B/W Image of 2 x 2 pixels and Colored Image of 2 x 2 pixels.

B/W Image is a 2D array, where every pixel has a value between 0 ≤ pixel value ≤ 255 (8 – bits of information). 0 is a completely white pixel and 255 is a completely black pixel.

Colored Image is 3D array. You got a blue layer, green layer and a red layer (RGB). Each color has its own intensity. Each pixel has 3 values associated with it and each value ranging from 0 ≤ pixel value ≤ 255.

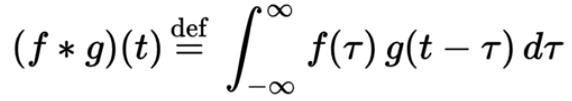


Below Image can be represented as a matrix of values –

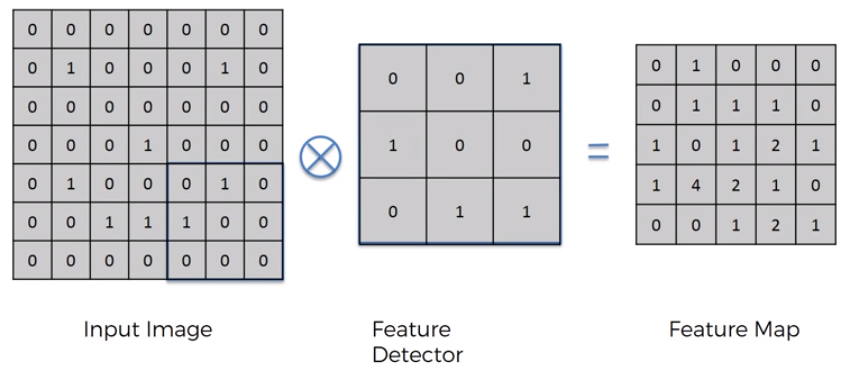


Step-1 : Convolution Operation

Below is the convolution function –



It is basically a combined integration of two functions and it shows you how one function modifies the shape of the another. Consider an input image in matrix form and a feature detector is a n X n matrix (Usually n = 3). N is different for different architecture. Also called as filter or kernel. Symbol for Convolution operation is - . You take feature detector and put it on the image, and multiply each value in position (m,n) by the value in feature detector at same position and then add everything. It is basically element wise matrix multiplication. The step at which we are moving the filter is called Stride. Here stride = 1 pixel. Stride can be changed. Conventionally stride is = 2 pixel.

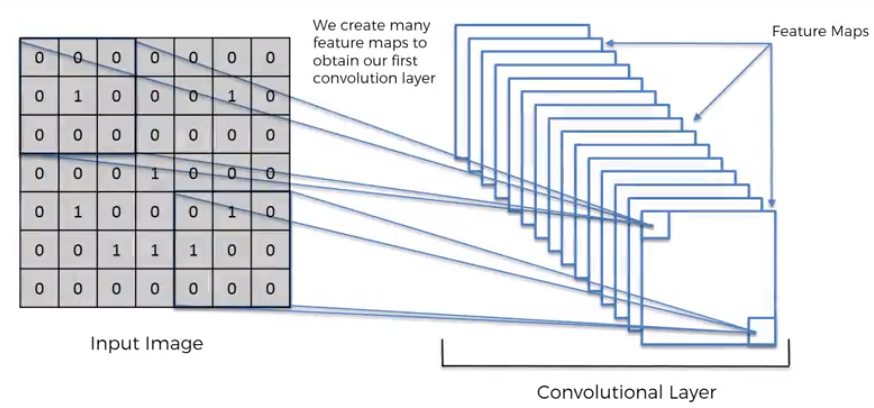


The feature map can be called Convolved feature or Feature Detector also. After applying convolution step -

1. We have reduced the size of the image. As the stride increases, image will be reduced more and feature map size decreases. So, the processing speed of image increases, and execution becomes faster.

Do we loose information after applying feature detector?

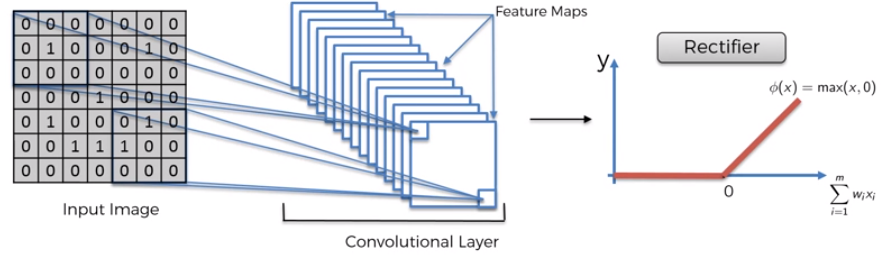
Yes, of course, we loose some info. But the purpose of feature detector is to identify certain parts of the image that is integral. We create multiple features maps using multiple filters to preserve the features through its training, it decides which features are important for certain categories and uses different filters. So, called as feature detector.



Primary use of convolution is find features in the image, collect them using feature detector, put them in feature map. It still preserves spatial relationship between pixels.

Step-1 (b) : ReLU Layer – Rectified Linear Units

Additional step after applying convolution layer. We apply rectifier function on feature maps.



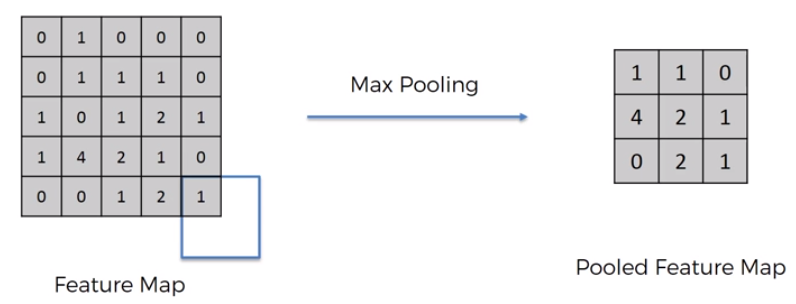
We use rectifier because we want to increase non-linearity in our image. Rectifier acts as a function that breaks up linearity. Why do we want to increase non-linearity in the images? Because, images themselves are highly non-linear. Image will have lots of non-linear elements in transition between adjacent pixels.

Step-2 : Max Pooling

Neural Network should be able to recognize the image even if it is expanded, rotated, squashed or if the image is taken from different angles, turned in different directions i.e. there are lots of minor differences. NN should recognize based on distinctive features. NN should have property called spatial invariance. If the feature is little distorted (slight differences relative to the training set), NN should have flexibility to be able to still find those features.

Consider feature map. We apply max pooling –

Find 2 x 2 boxes, and keep the maximum value in the box. Stride = 2 pixels

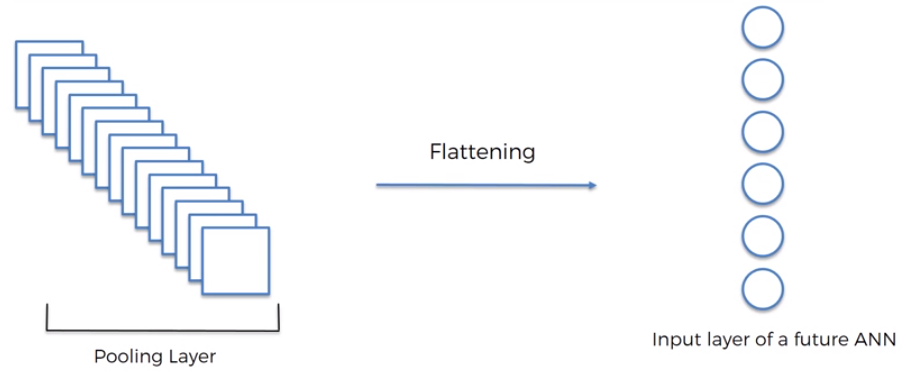


We still are able to preserve the features and account for possible spatial, textural and other distortions. The maximum number in the feature map is where we found out close similarity to the features. By pooling these features, we are getting rid 75% of the information which is not the feature. We are reducing the size by 75% to facilitate faster processing. We are also reducing the number of parameters that goes to the final layers of NN and hence avoiding overfitting. It is important to see the features rather than the noise.

Different types of pooling - Sub- Sampling is average pooling, sum pooling, min pooling

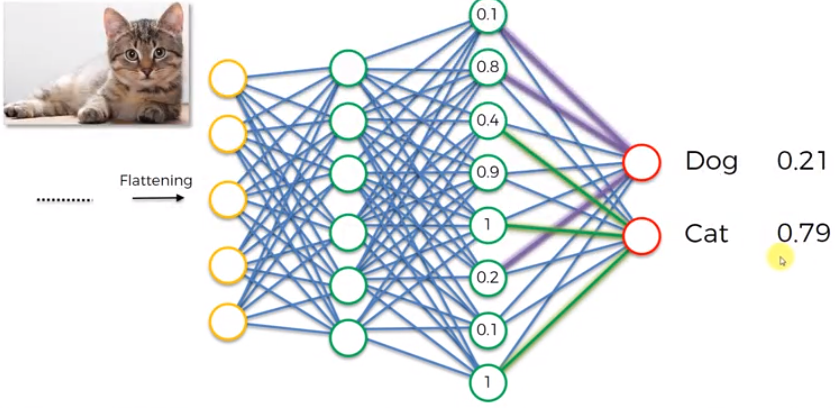
Step-3 : Flattening

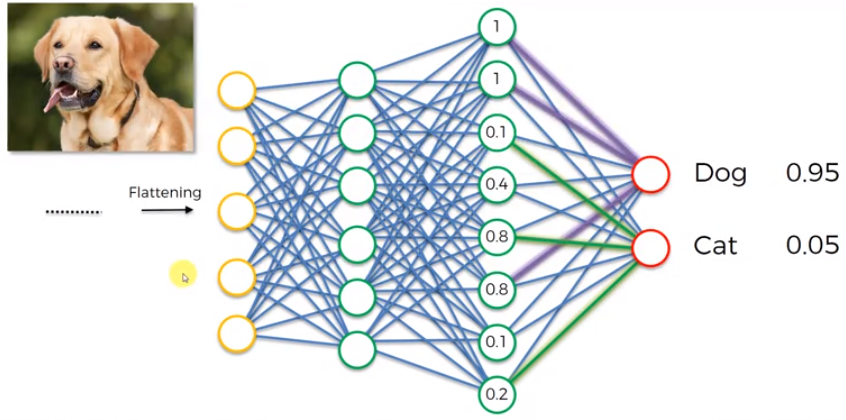
You take the pooled feature and flatten it into a column – Take numbers row aby row and put them into a long column sequentially one after another and you get a huge vector of inputs for an ANN. Because you later input it into an ANN for further processing.



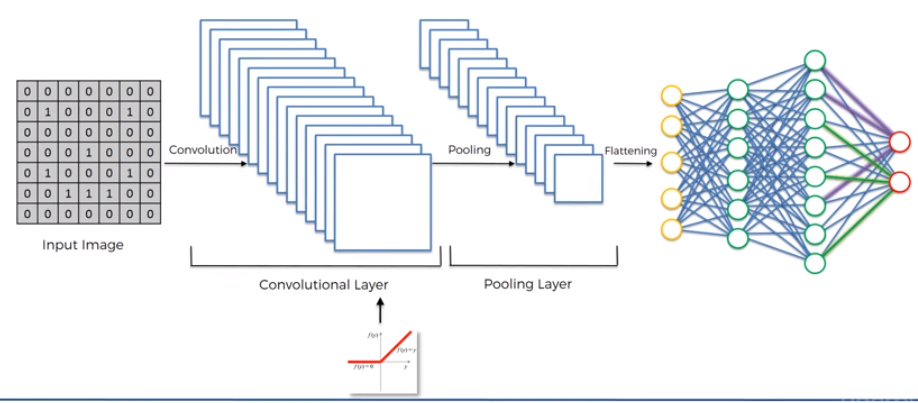
Step-4 : Full Connection

Adding a whole ANN to CNN in this step. Here, we have input layer, fully connected layer (hidden layers) and output layer. In ANN, the hidden layers don’t have to be fully connected but in CNN, hidden layers have to fully connected. Main purpose of ANN – Combine the features into more attributes that predict the classes even better. The vector of inputs that we give to ANN already encodes some features. In classification, if we have more than one category, then we need to have neuron per every category. Here, in error back propagation step in ANN, the weights as well as the feature detectors are adjusted. In the last hidden layer, for every category different set of neurons light up or get activated and depending on that, the output neuron predicts the category. That’s is how the features are propagated through the network and conveyed to the output neuron.



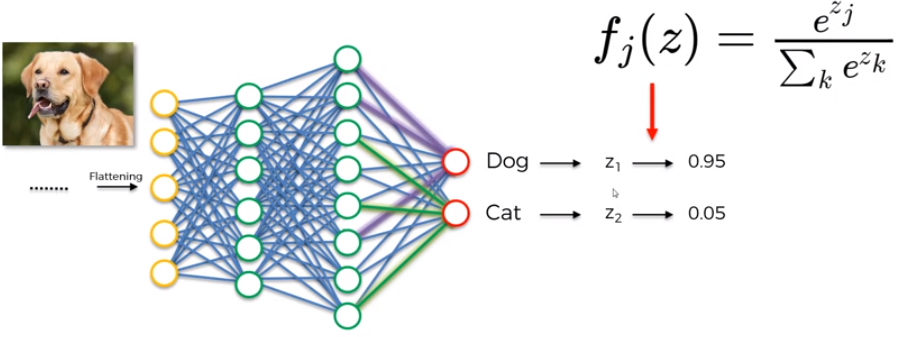


Summary :

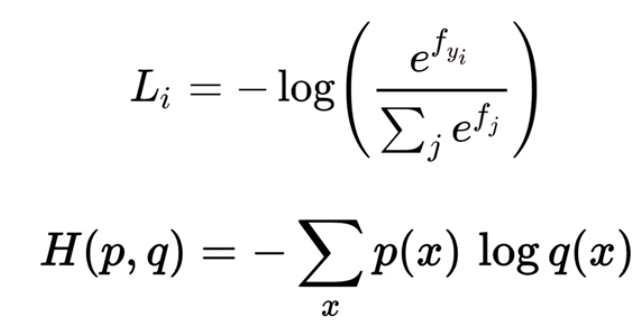


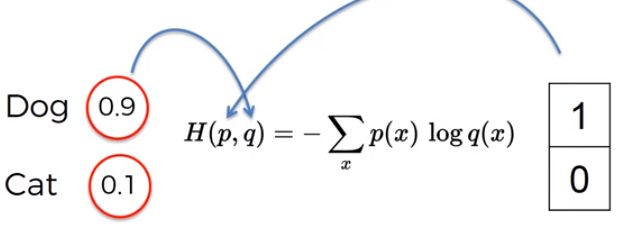
SoftMax and Cross-Entropy :

The output neuron in ANN gives out the probability of each of the classes that is being classified. Normally, these won’t add up to 1, it can have any real values. But by applying SoftMax function, it constraints the output to add up to 1 and brings each value between 0 ≤ Z1 ≤ 1.



Cross- Entropy function is the loss function in CNN- We minimize the loss function to maximize the performance of the network.





Advantages of Cross-entropy over Mean Squared Error –

1. If the output value is very very tiny and much smaller than actual value that you want. Then at the very start the gradient in the gradient descent will be very low and it will be very hard for neural network to actually start doing something and start adjusting weights and moving in the right direction. If we use cross entropy, since it has a logarithm, it will easy to assess such small errors. It is a better way for NN to get to the optimal state.

Reading Materials -

9 Deep Learning Papers to understand CNN by Adit Deshpande – click [here](https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html)

A friendly Introduction to [Cross Entropy Loss](https://rdipietro.github.io/friendly-intro-to-cross-entropy-loss/)

Implementing [Neural Network - Intermezzo](peterroelants.github.io/posts/neural_network_implementation_intermezzo02/)