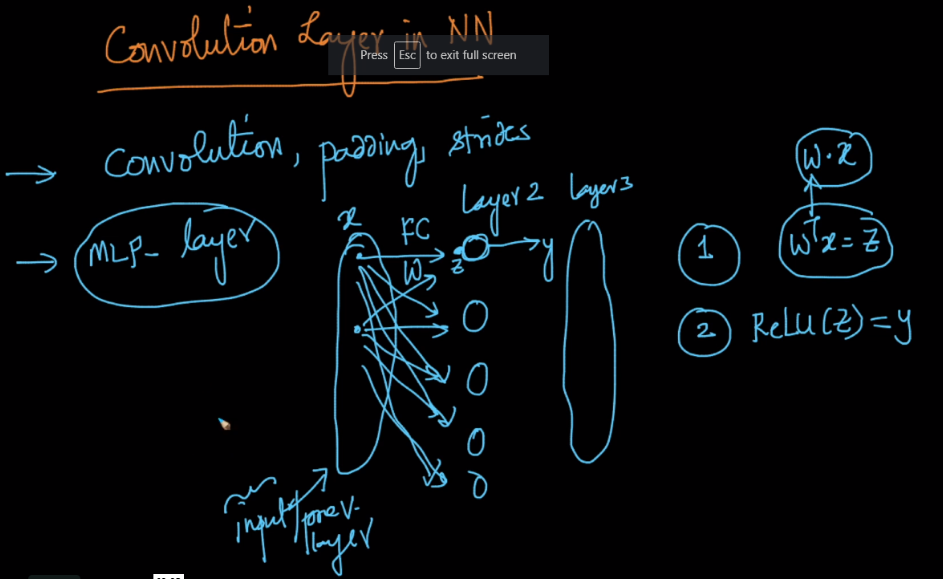
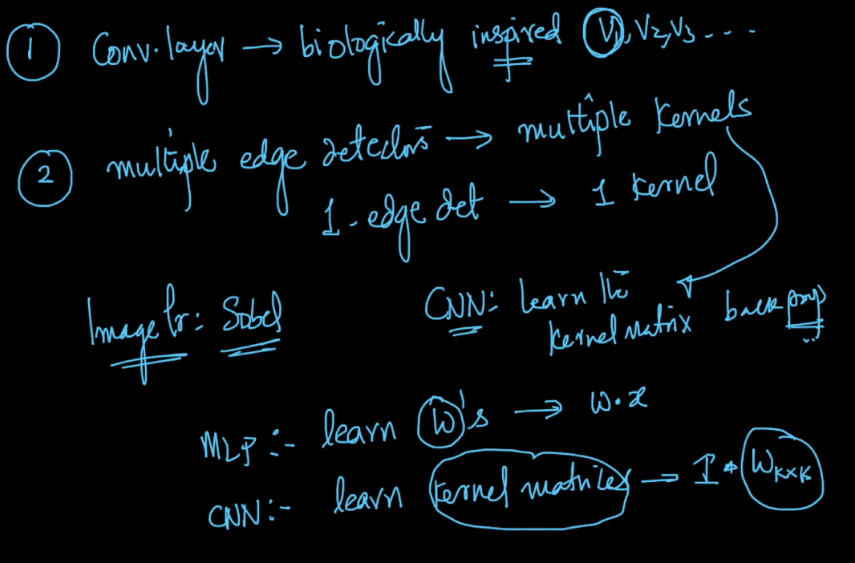
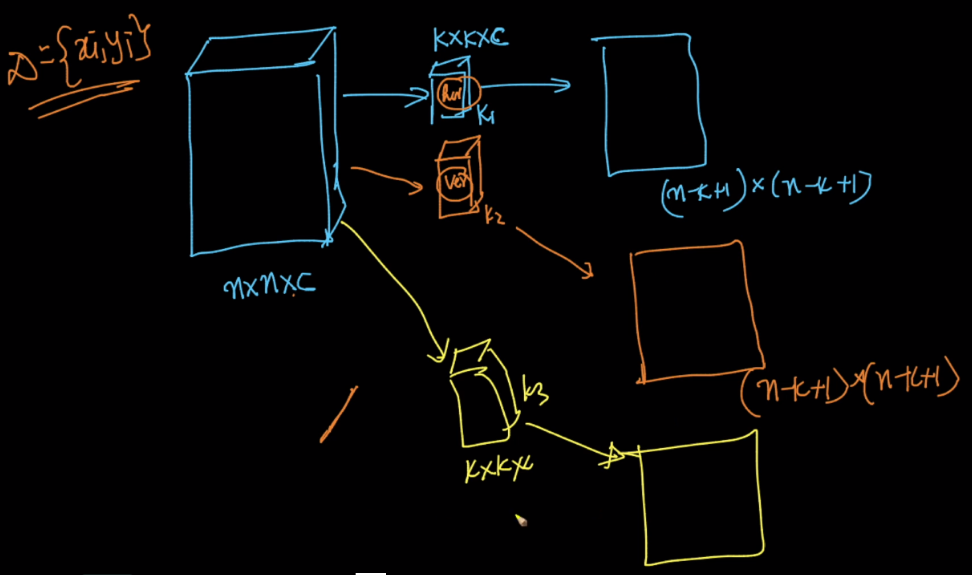
As we know in MLP we have many layers and is a fully connected network, where in each layer we apply dot product on i/p and weight and then apply activation function on that.

In CNN also we have many layers but the network will not be fully connected it will be sparse, why it has many layers, because suppose first layer is used to detect edges, second used to detect regions etc.

Architecture for CNN:

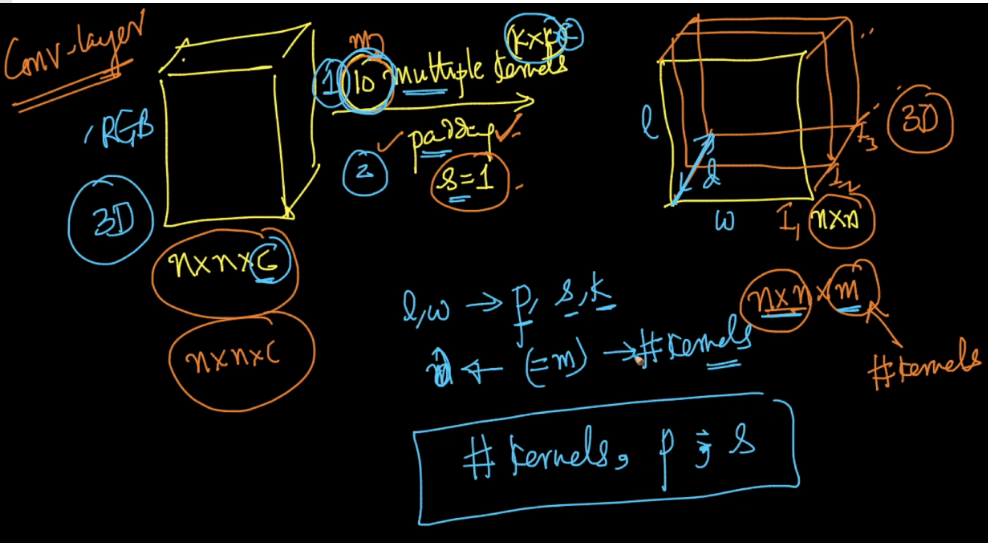
We have a i/p RGB image, now this will be passed through different kernels, where each kernel is responsible for detecting different things(ex edges at different angles), then from each kernel we’ll get a convolued matrix.



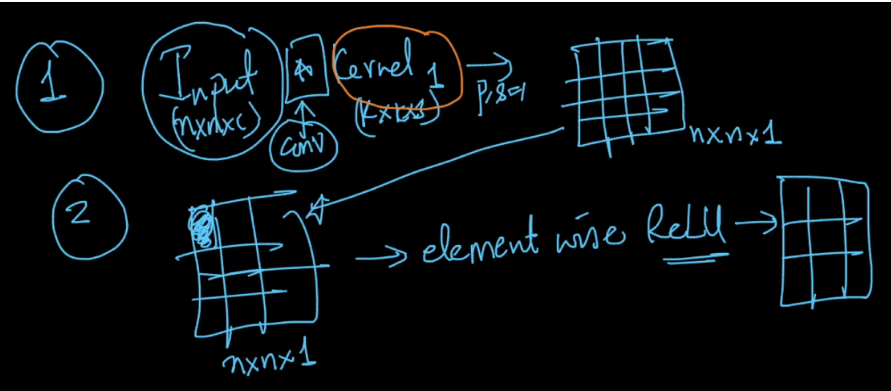
So let’s say i/p is a RGB image of n\*n dim, so input will be of n\*n\*3,

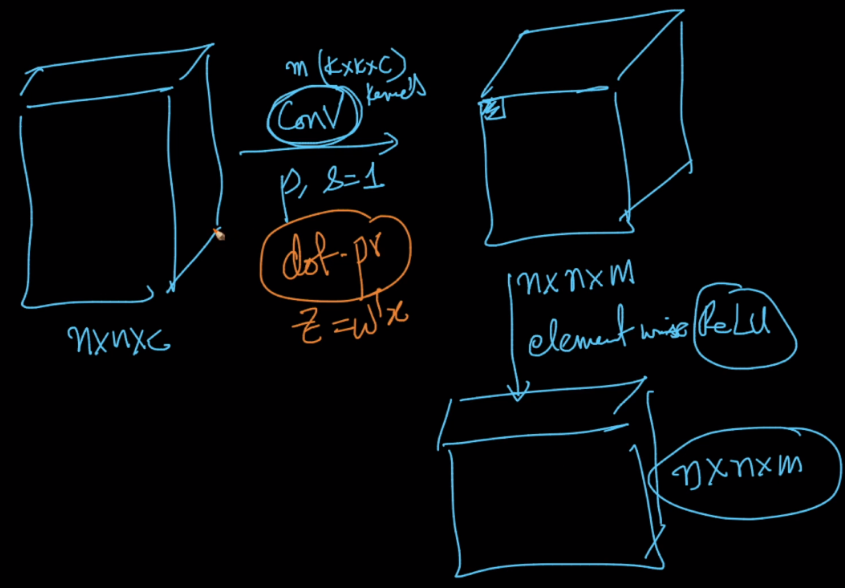
Now those will be given to m kernels of size k\*k with proper padding and stride=1.

From kernel layers, it will generate ‘m’(no. of kernels) matrix of size n\*n

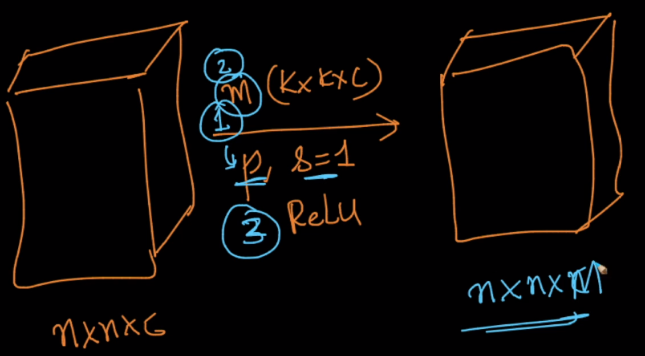


After getting output matrix from kernel layer, now each cell of matrix is passed to RELU or activation function.

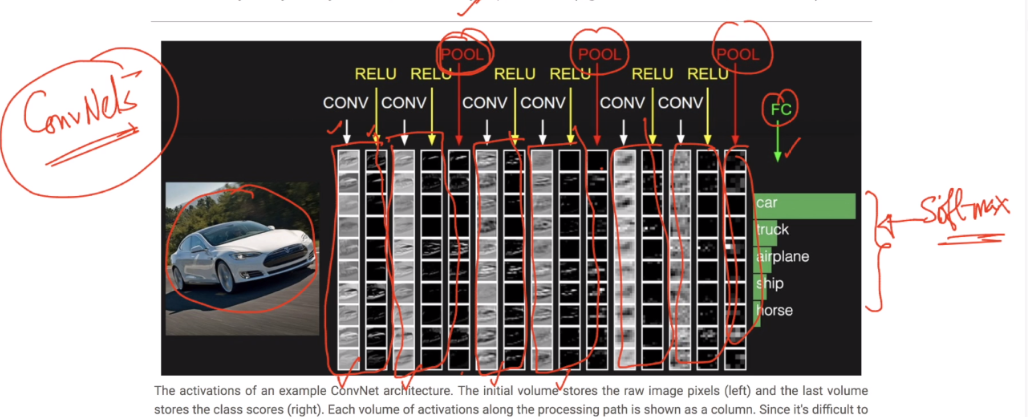


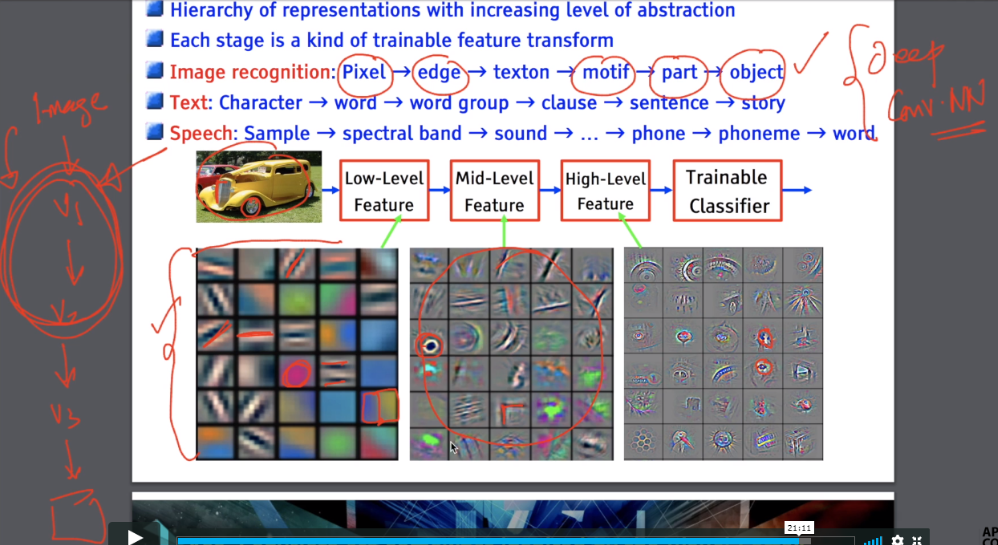


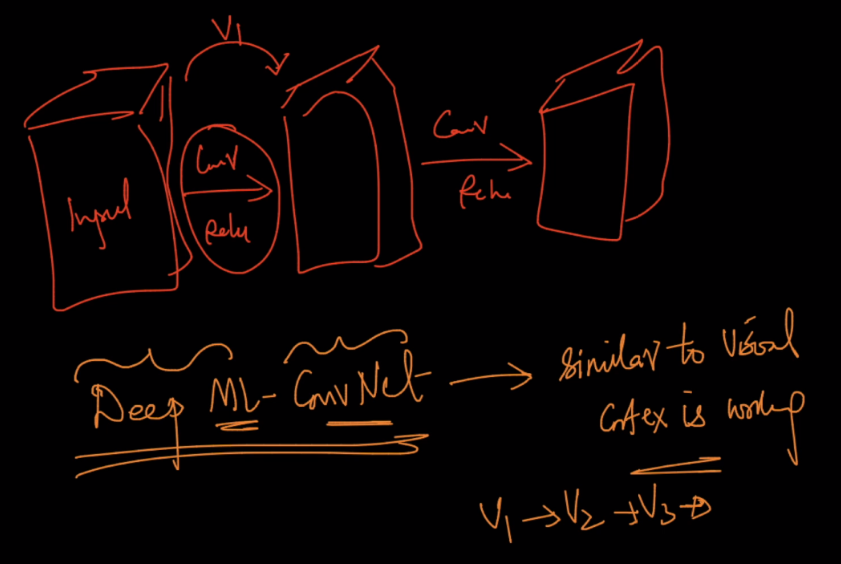
Often these 3 layers of i/p, kernel and relu are written as 2 layers which is shown below.

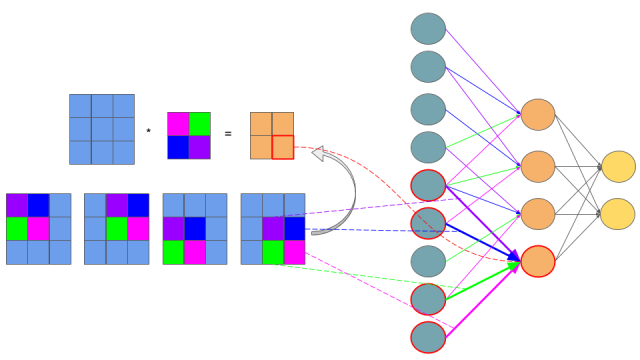


Example of convNets which consist of multiple conv+Relu layers.









<http://cs231n.github.io/convolutional-networks/#pool>

<https://www.quora.com/What-is-the-role-of-rectified-linear-ReLU-activation-function-in-CNN>

