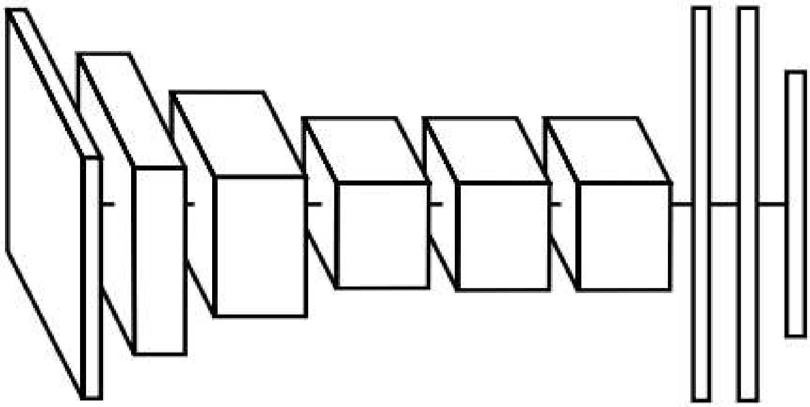
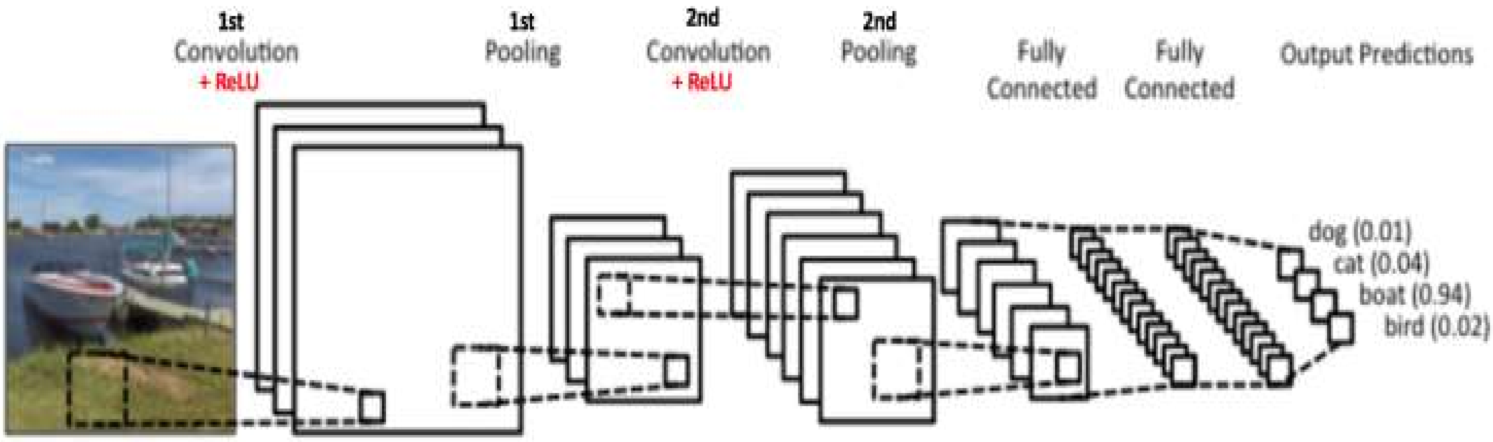
Convolutional Neural Network(CNN)



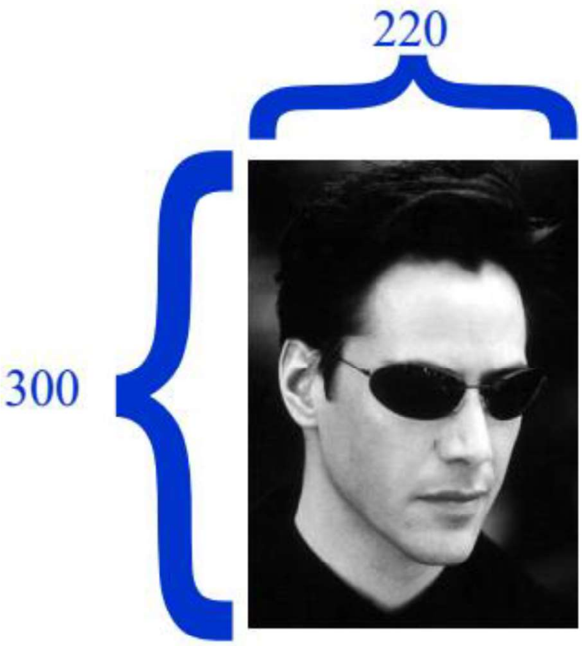
Our goal: recognize the image as boat



What is an image?

Grey image

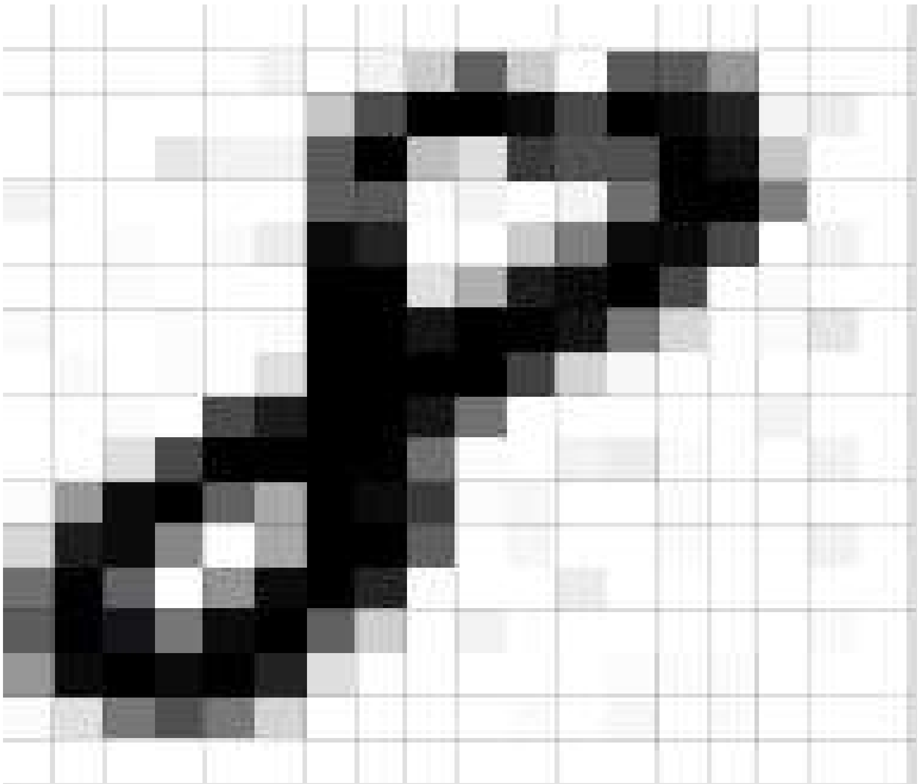
- A grid of numbers (*intensity values*)
- In computing, a *matrix*



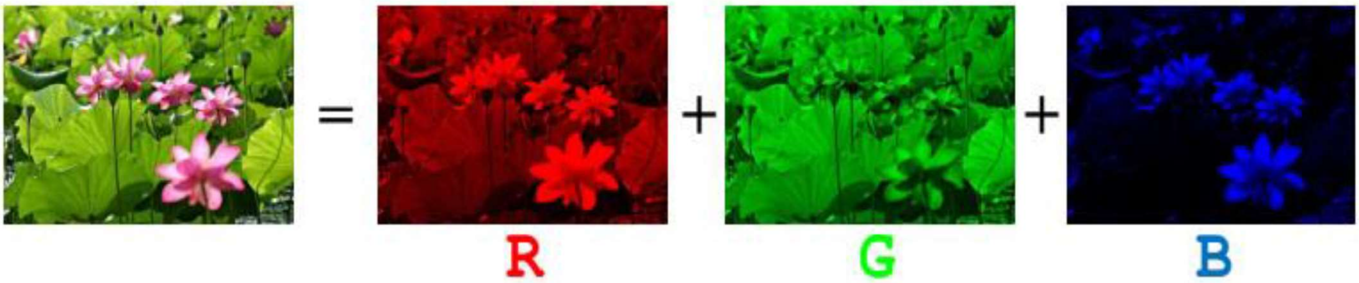

```
[ 10  30  40  106  123  ...  ;
   8  49  58  112  145  ...  ;
  16  53  86  123  152  ...  ;
                ...              ]
```

300 x 220 matrix

Digit 8 in grey scale



Color image



Convolution

Extracting features from images

- Extract features by applying a sliding window function across image, called as convolution

- Why sliding window?
  - Detect the same feature at different positions in the input image

How do you apply convolution across image?

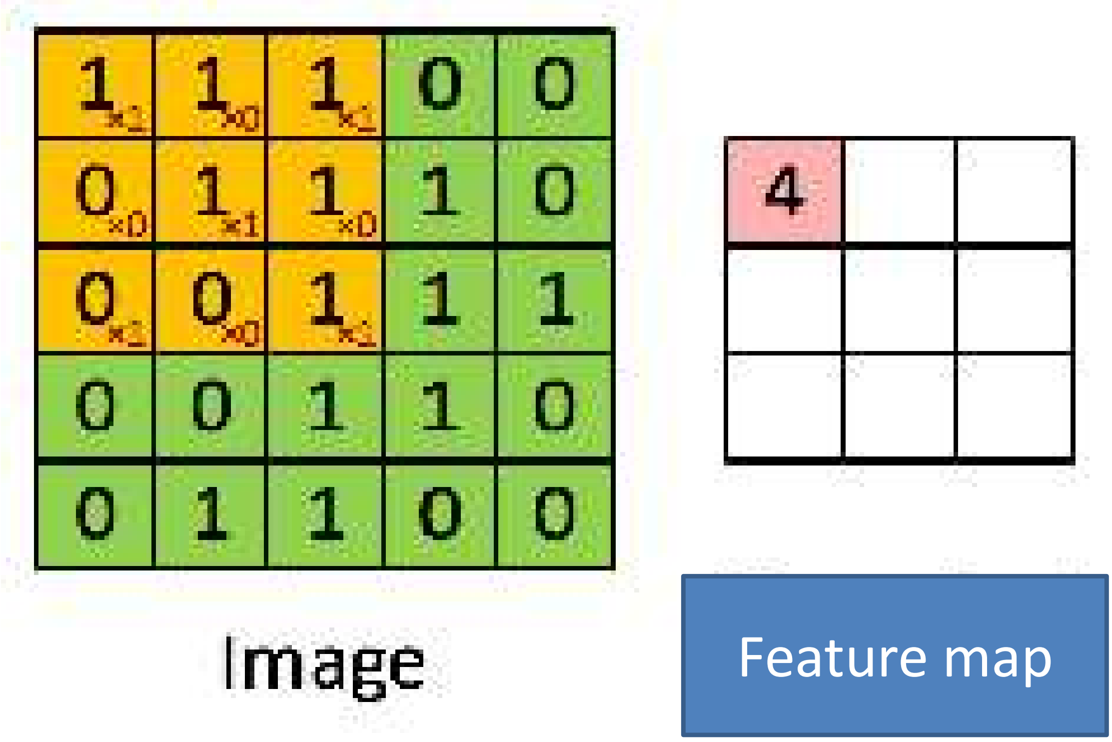
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

image

1	0	1
0	1	0
1	0	1

Convolutional filter

How do you apply convolution across image?



Example1

Different filters (weights) reveal a different characteristics of the input.

$$* \, 1/8 \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Example1

Different filters (weights) reveal a different characteristics of the input.

$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Different filters (weights) reveal a different characteristics of the input.

$$* \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

Example2: Feature maps

- The convolution of a filter1 (with red outline) slides over the input image to produce a feature map.

- The convolution of another filter2 (with the green outline), over the same image gives a different feature map

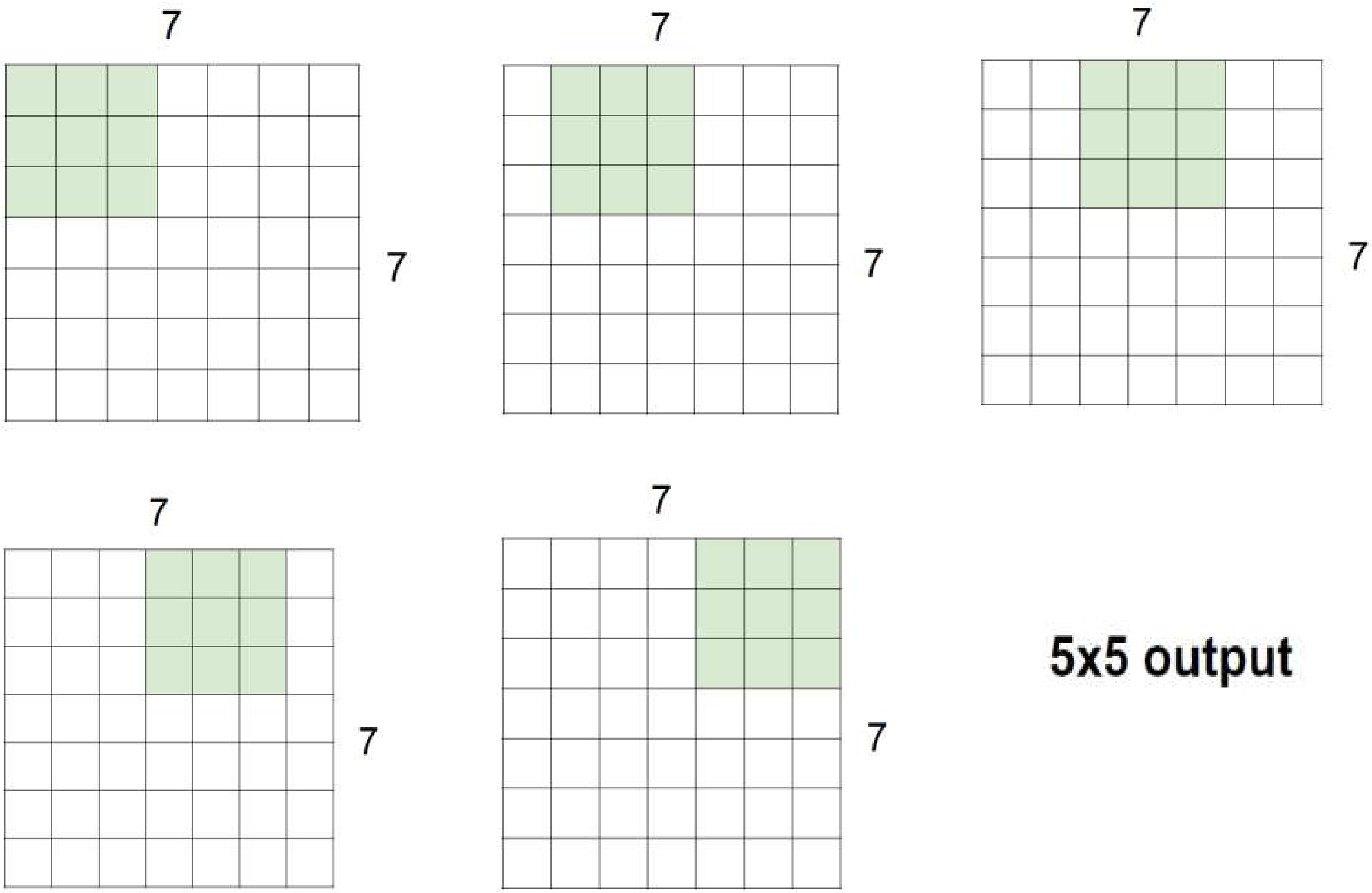
Parameters for convolutional filters

Parameters for convolutional filters

The size of the feature maps are controlled by following parameters of convolution filters:

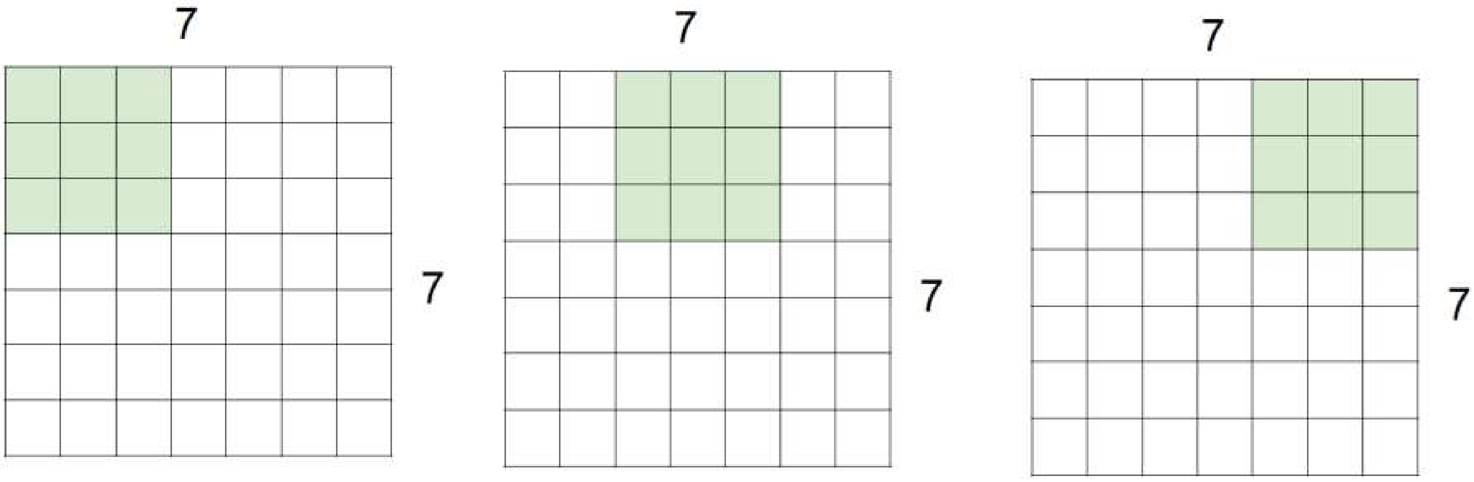
- Depth: It corresponds to the number of filters we use for the convolution operation
- Stride: It is the number of pixels by which we slide our filter matrix over the input matrix
- Padding: Adding zeros around the border, so that we can apply the filter to bordering elements of our input image matrix

7 by 7 input, 3 by 3 filter with stride 1



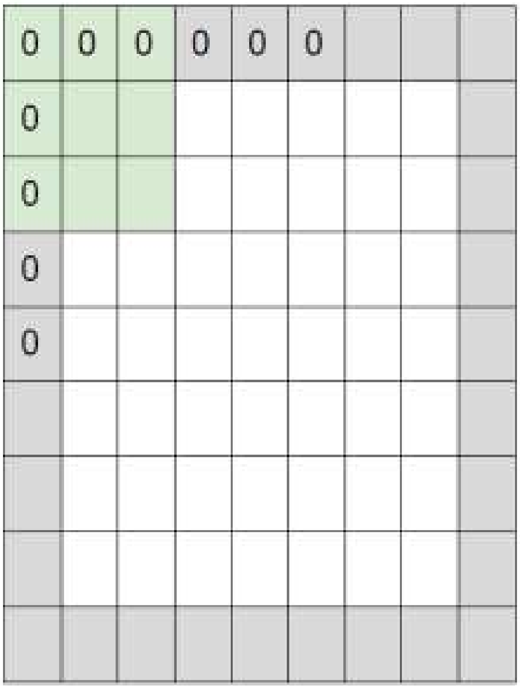
5x5 output

7 by 7 input, 3 by 3 filter with stride 2



3x3 output

Convolution for edge cases: zero pad the border



7 by 7 input, 3 by 3 filter with stride 1, pad with 1 pixel border

7 by 7 output

Generalization: zero pad the border

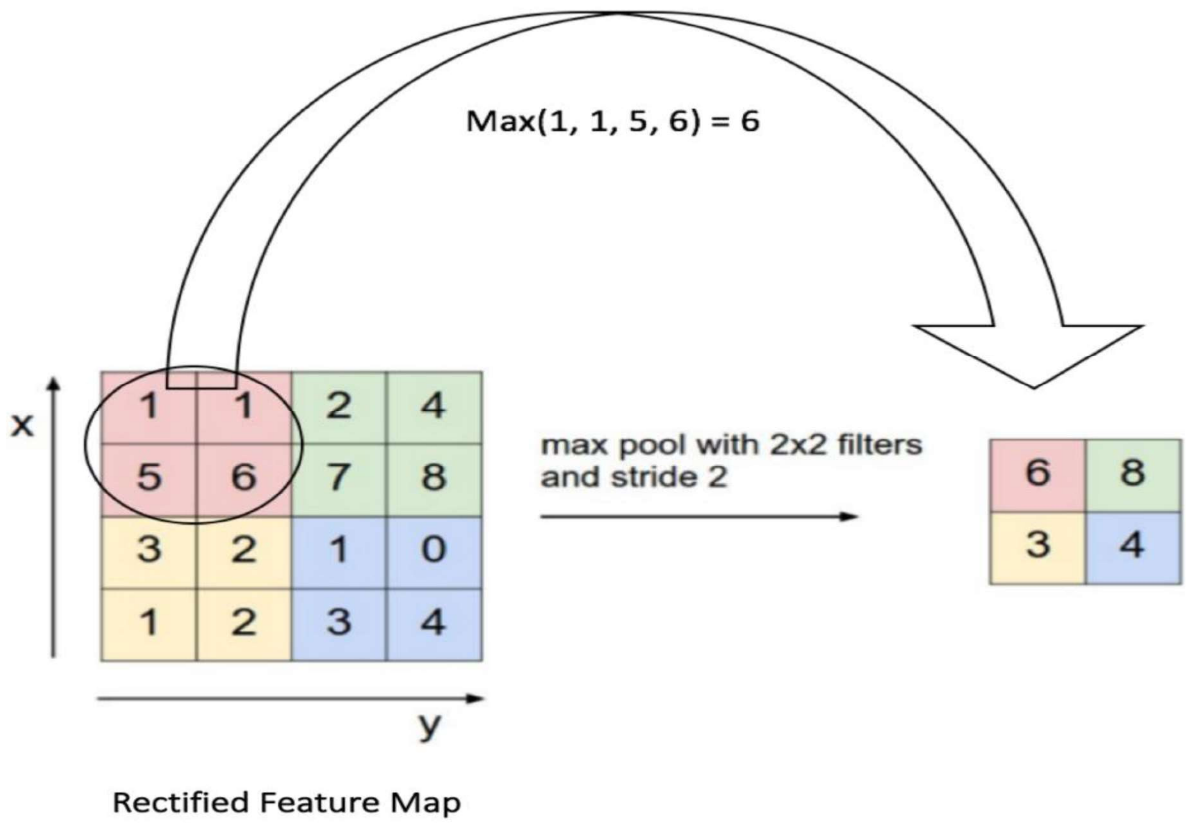
- In general, convolution layer will be having filters of size F by F with stride 1. The number of zero pad borders will be (F-1)/2 to preserve size spatially

- E.g.,

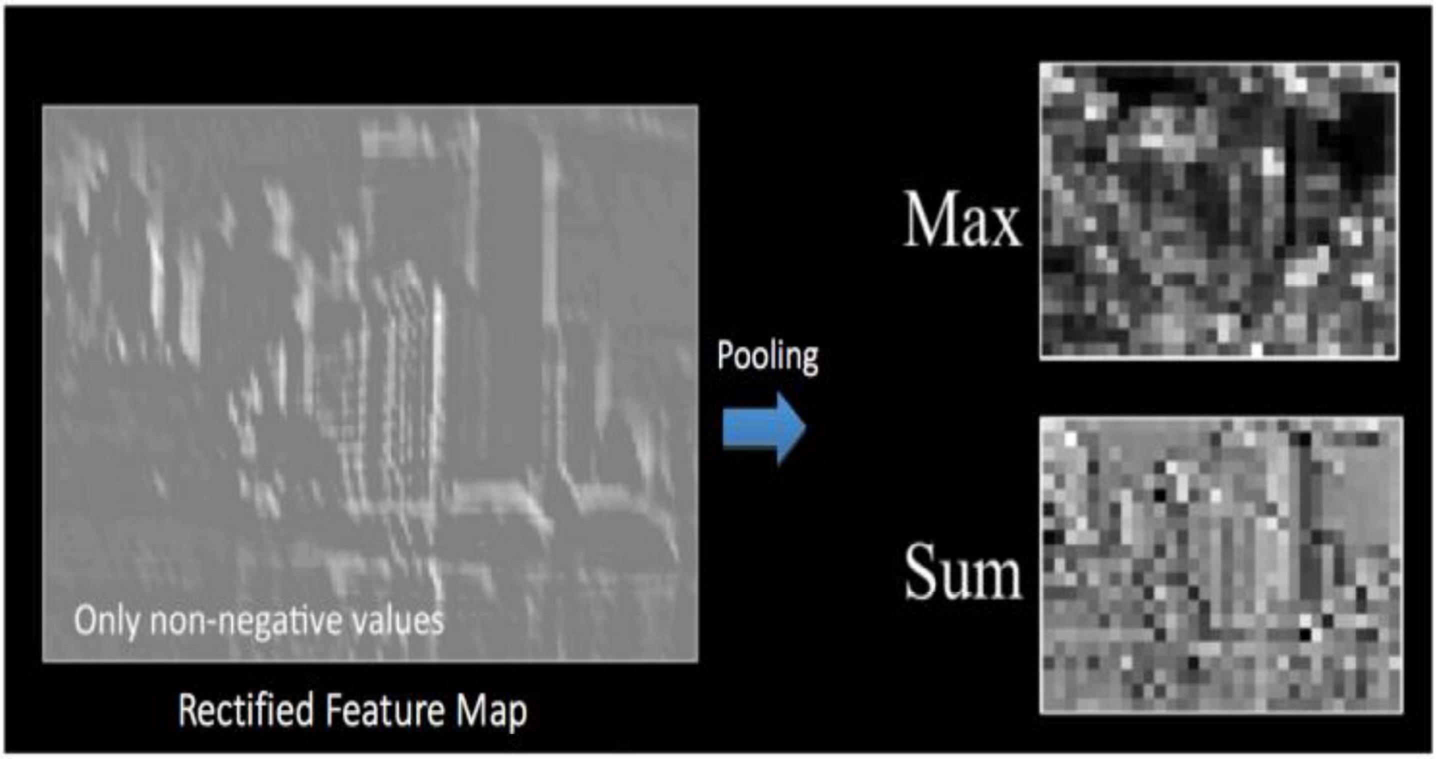
  F = 3 => zero pad with 1
  F = 5 => zero pad with 2
  F = 7 => zero pad with 3

Pooling/Subsampling

Max pooling


Max(1, 1, 5, 6) = 6
x
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
y
Rectified Feature Map
max pool with 2x2 filters
and stride 2
6 8
3 4

Max pooling


Only non-negative values
Rectified Feature Map
Pooling
Max
Sum

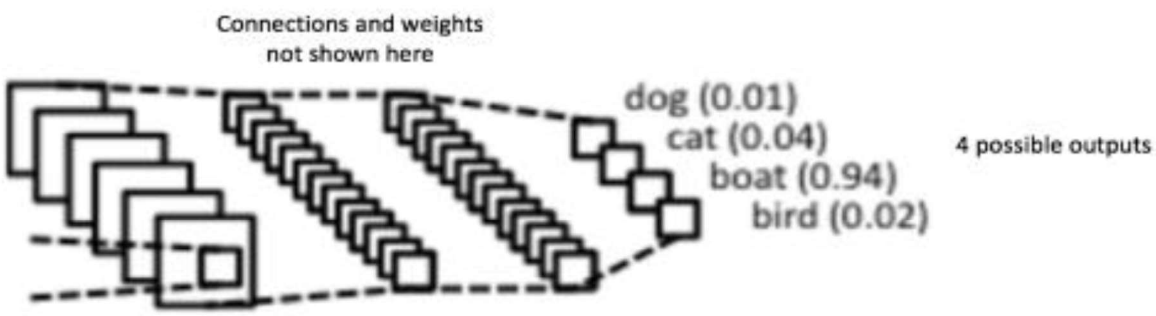
What does pooling gives us?

- Makes the feature dimension smaller and fixed size

- Invariance to small transformations i.e., reduce the effect of noises and shift or distortion

Fully connected layers

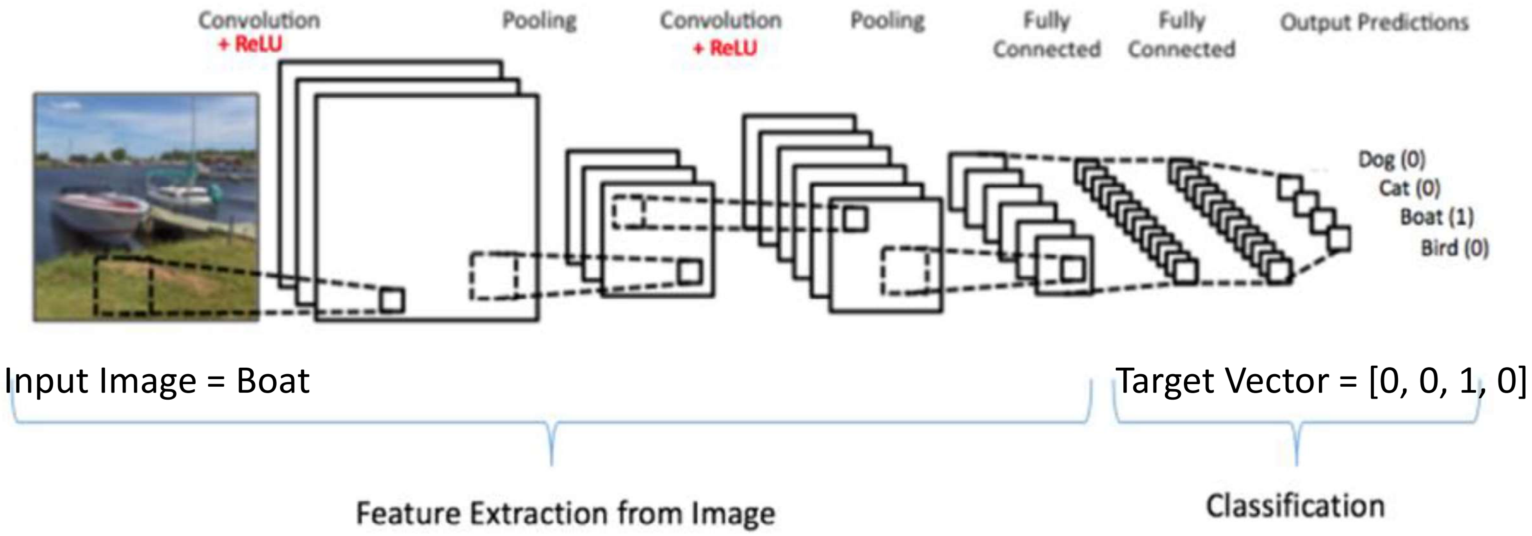
FC layer



- The output from the convolutional and pooling layers represent high-level features of the input image

- The purpose of the Fully Connected layer is to use these features for classifying the input image into various classes based on the training dataset

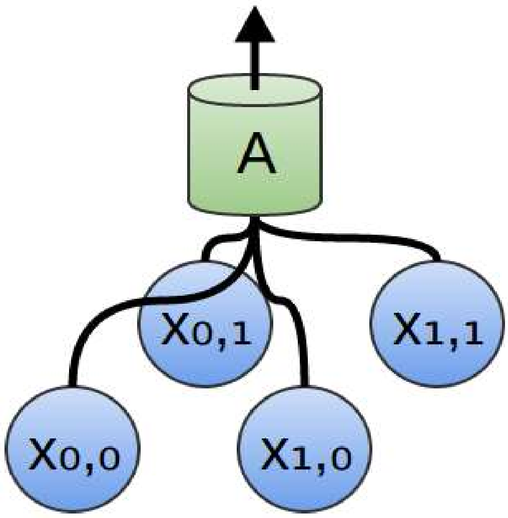
The final convolutional neural network



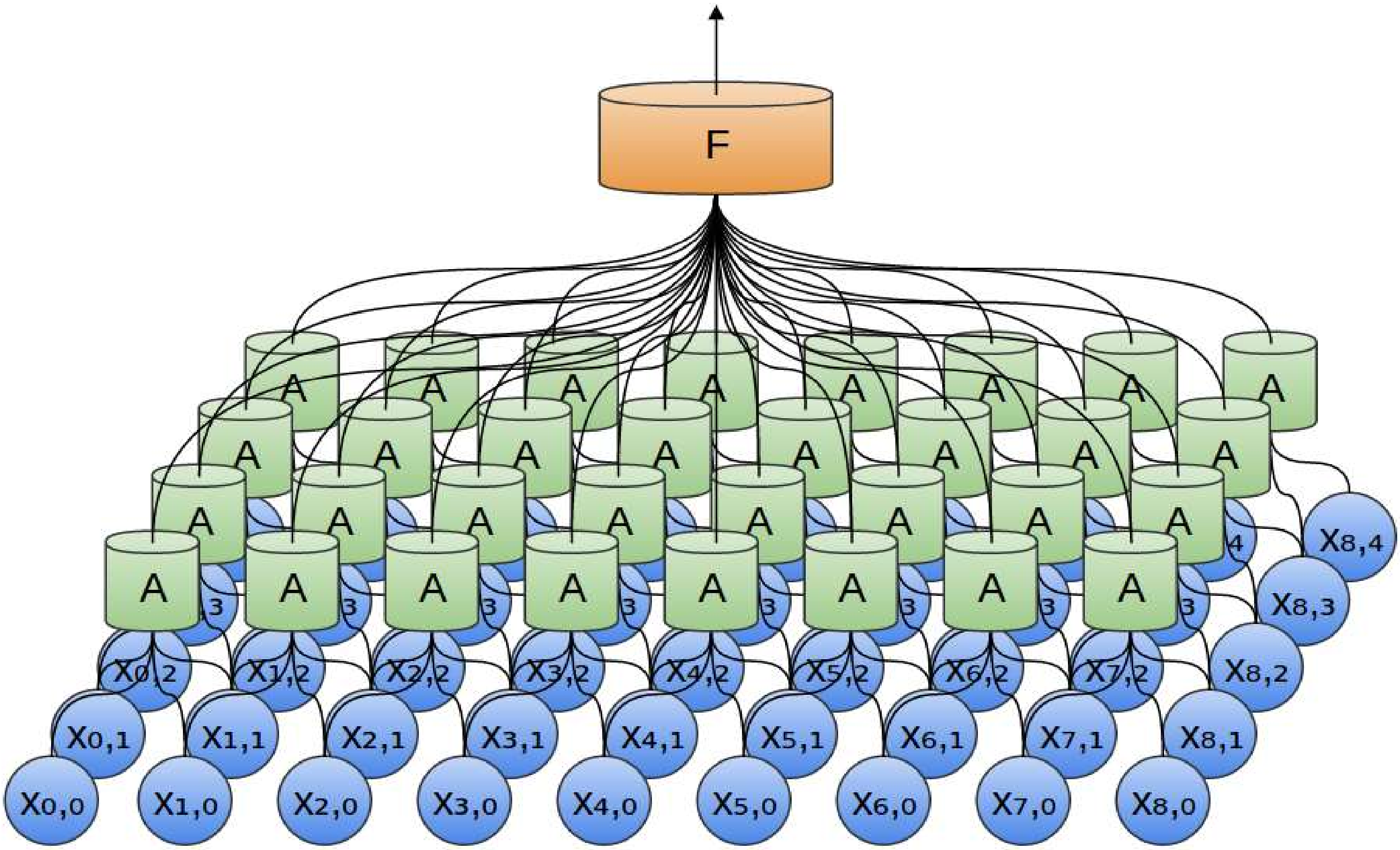
The Convolution + Pooling layers act as Feature Extractors from the input image while Fully Connected layer acts as a classifier.

Perceptron view of CNN

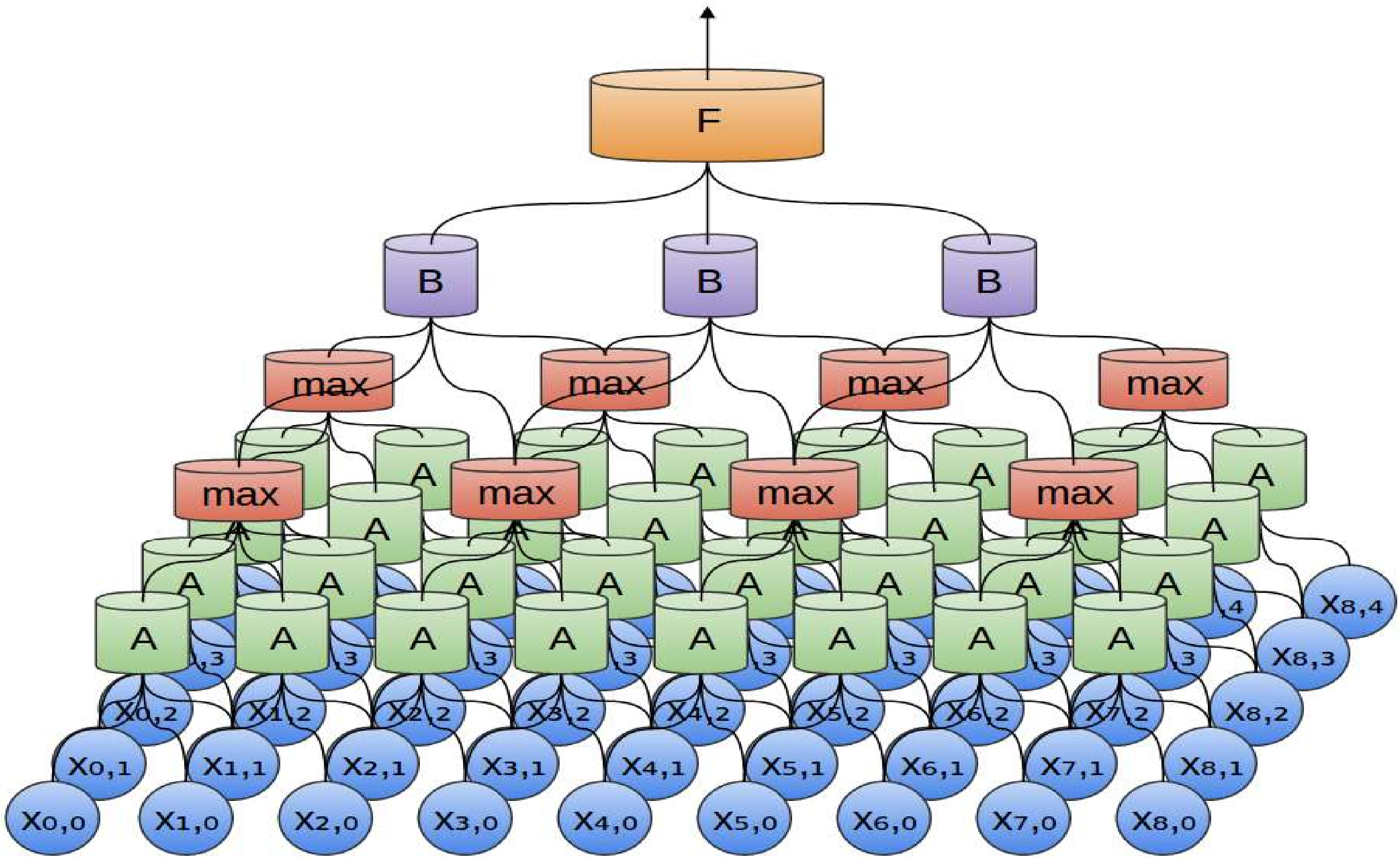
Convolution Filter = replicated perceptrons



Convolutional Filter = replicated perceptrons



Conv filter + Maxpooling + Conv filter + FC

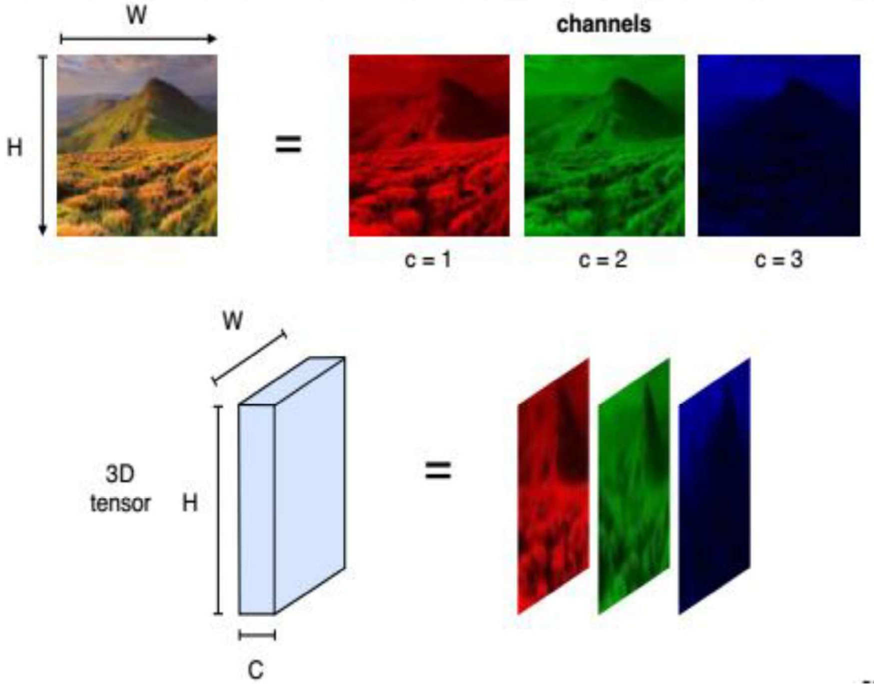


Channels & Activation maps

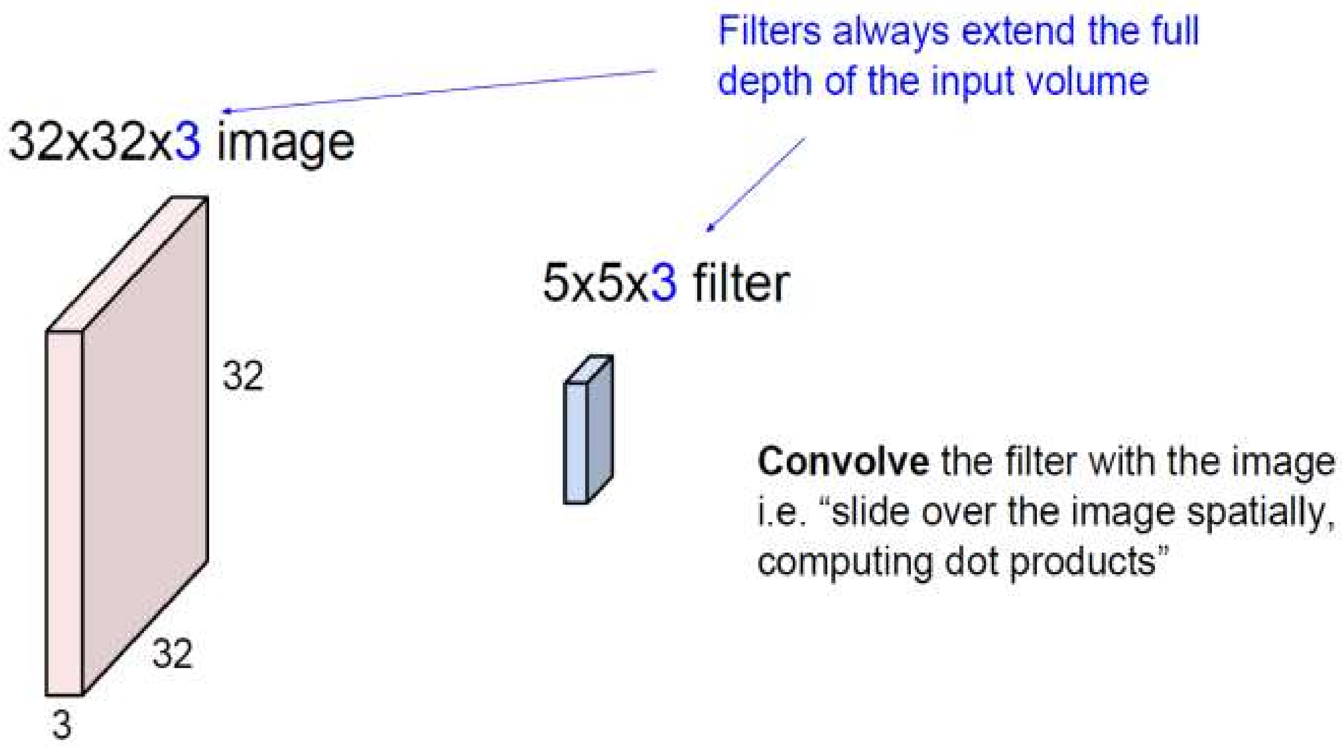
3D Data

Data = 3D Tensor

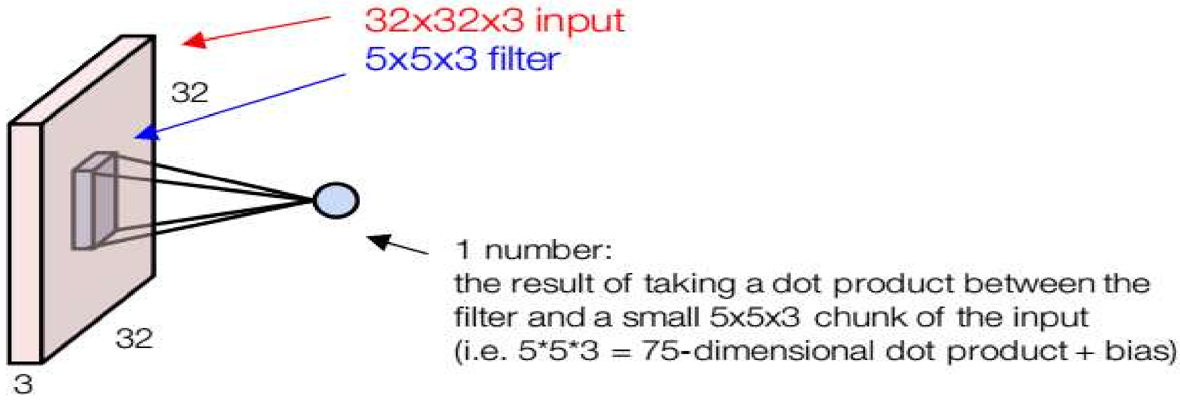
- There is a vector of feature channels (e.g. RGB) at each spatial location (pixel).

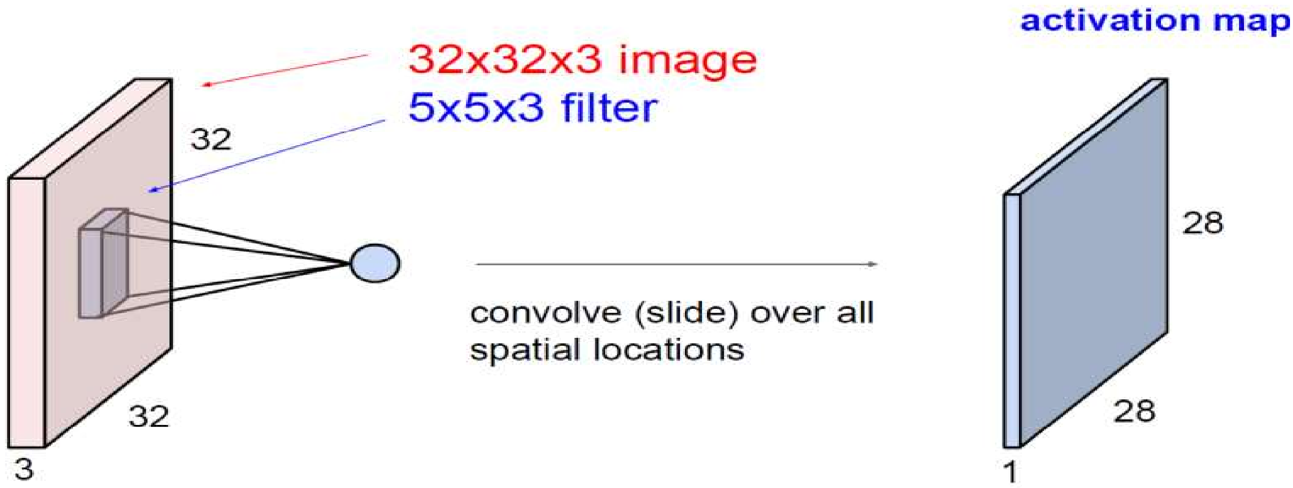


Convolution with 3D Filters

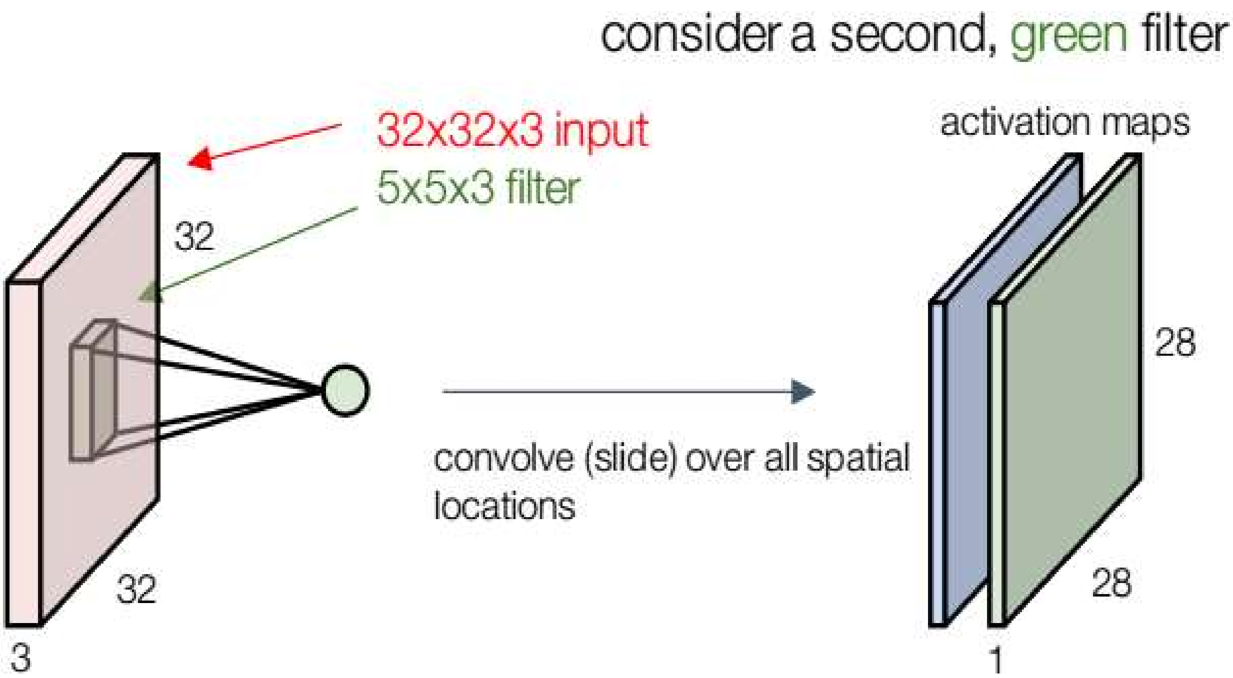


Convolution with 3D Filters



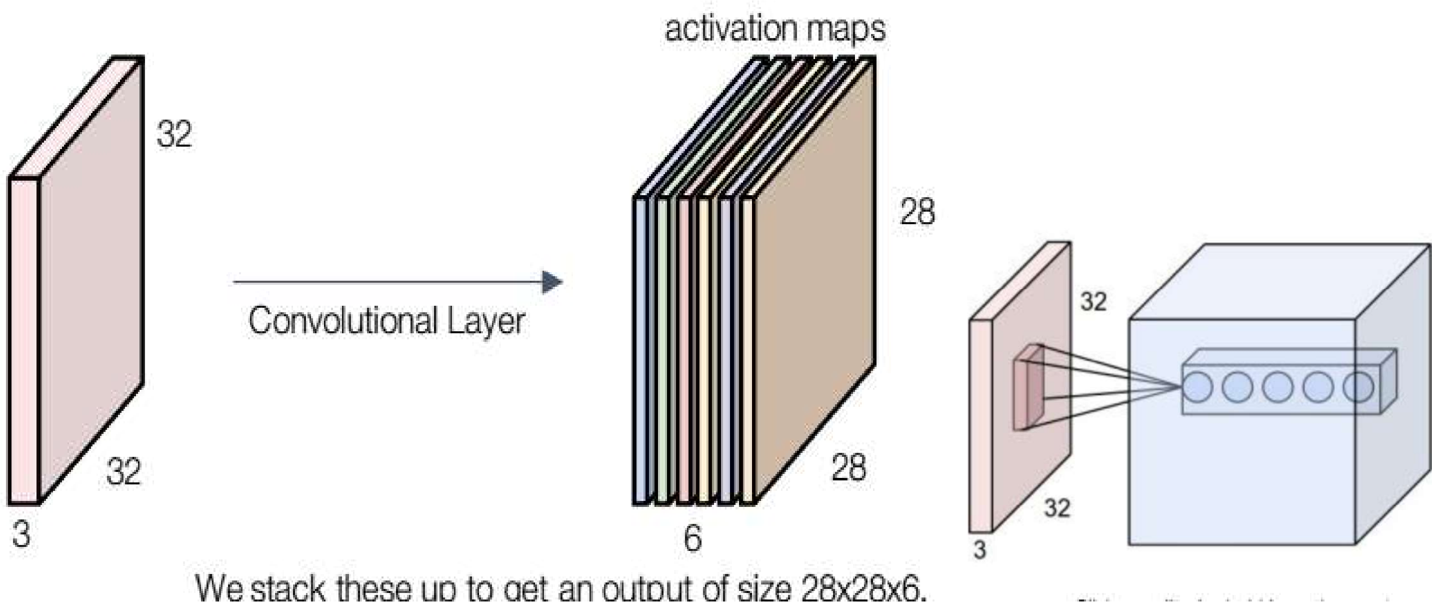


Convolution with 3D Filters



Convolution with 3D Filters

- Multiple filters produce multiple output channels
- For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



Pooling across feature maps

- makes the representations smaller and more manageable
- operates over each activation map independently: