CSE 4621
Machine Learning

Lecture 10

Md. Hasanul Kabir, PhD.

Professor, CSE Department

Islamic University of Technology (IUT)

deeplearning.ai

Convolutional Neural Networks

Introduction

Source & Special Thanks to (Coursera) CNN Course (Deep Learning Specialization)

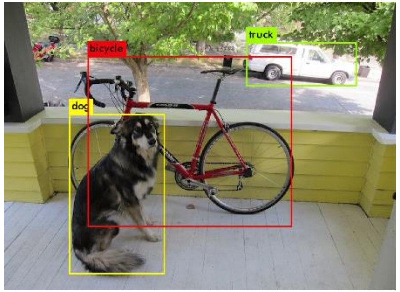
Computer Vision Problems

Image Classification

64x64

Cat? (0/1)

Object Detection



Neural Style Transfer

Deep Learning on large images

64x64×3

→ Cat? (0/1)

12288

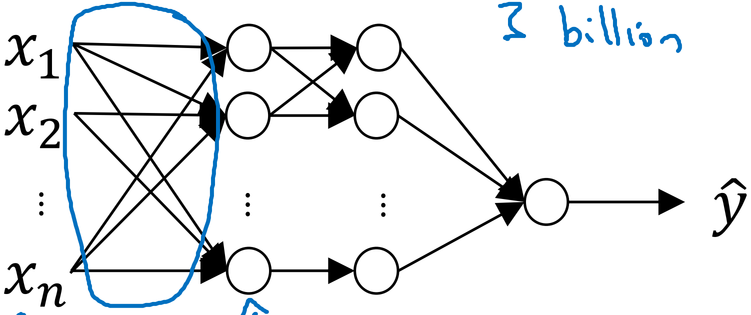
$1000 \times 1000 \times 3$
$= 3$ million

- Learning 3 billion parameters for just one layer is too computationally expensive.
- **Convolution layers** provide solution to this problem.

$x \in \mathbb{R}^{3M}$

$W^{[1]}$ (1000, 3M)

3 billion



↑ 3M ↑ 1000

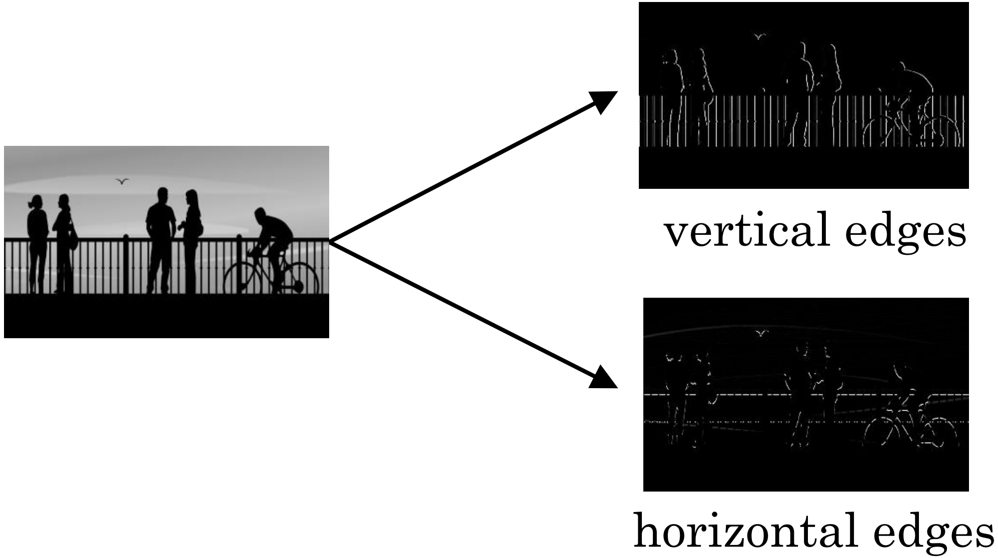
deeplearning.ai

Convolutional Neural Networks

Edge detection with Convolution

Feature Extraction in Computer Vision

Edge detection is a basic example of **convolution** operation that is a fundamental element in the **convolution layers**.



Convolution Operation (Step1)

Input

4	9	2	5	8	3
5	6	2	4	0	3
2	4	5	4	5	2
5	6	5	4	7	8
5	7	7	9	2	1
5	8	5	3	8	4

$n_H \, x \, n_W = 6 \, x \, 6$

$*$

Filter

1	0	-1
1	0	-1
1	0	-1

Parameters:

Size: $f = 3$
Stride: $s = 1$
Padding: $p = 0$

=

Result

2			

$\boxed{2}$ = 4*1 + 9*0 + 2*(-1) +
5*1 + 6*0 + 2*(-1) +
2*1 + 4*0 + 5*(-1)

https://indoml.com

Convolution Operation (Step 2)

Input

4	9	2	5	8	3
6	2	4	0	3	
2	4	5	4	5	2
5	6	5	4	7	8
5	7	7	9	2	1
5	8	5	3	8	4

$n_H \, x \, n_W = 6 \, x \, 6$

$*$

Filter

1	0	-1
1	0	-1
1	0	-1

Parameters:

Size: $f = 3$

Stride: $s = 1$

Padding: $p = 0$

$=$

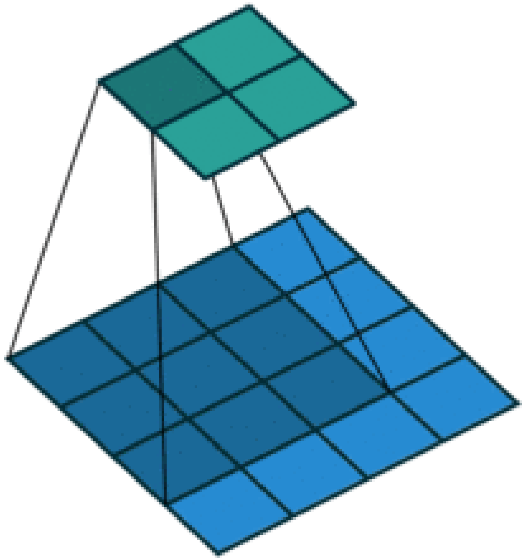
Result

2	$\boxed{6}$		

$\boxed{6}$ = 9*1 + 2*0 + 5*(-1) +
6*1 + 2*0 + 4*(-1) +
4*1 + 5*0 + 4*(-1)

https://indoml.com

Why Convolution Operation?



- **Parameter sharing**: A kernel is shared among every section of the input. For example, an edge detector is useful in detecting edges at any part of the image, with just few numbers.
- **Sparsity of connections**: each element of the output depends only on the small section of the input.

Vertical edge detection

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

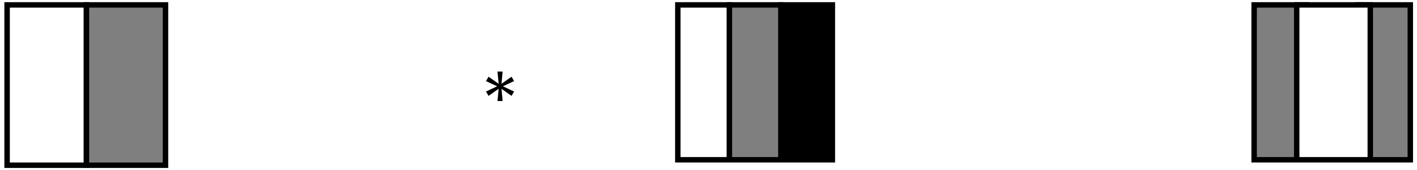
*

=

-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

Vertical edge detection

$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}$$



Goal: Learning to detect edges

1	0	-1
1	0	-1
1	0	-1

1	0	-1
2	0	-2
1	0	-1

Sobel filter

3	0	-3
10	0	-10
3	0	-3

Scharr filter

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

convolution

$*$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

3×3

=

45°
70°
73°

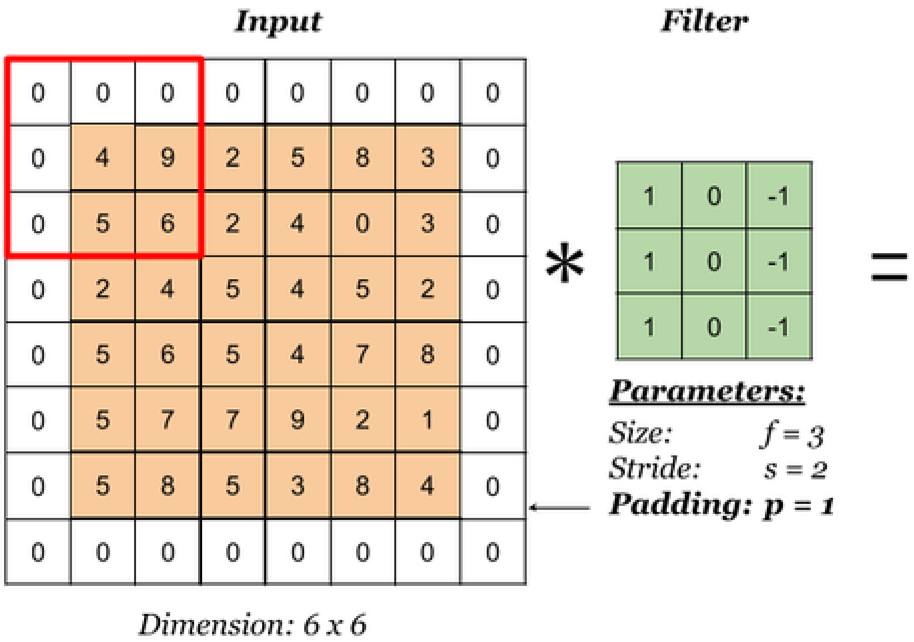
deeplearning.ai

Convolutional Neural Networks

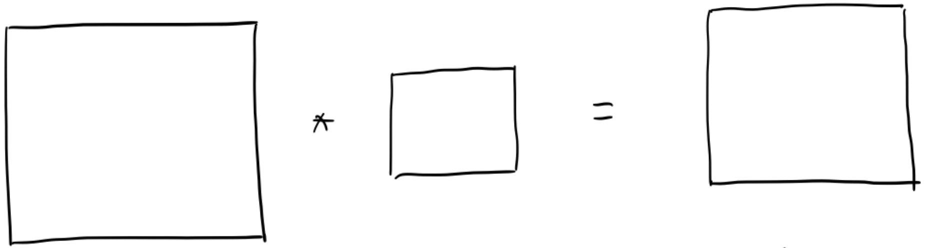
Padding

Padding

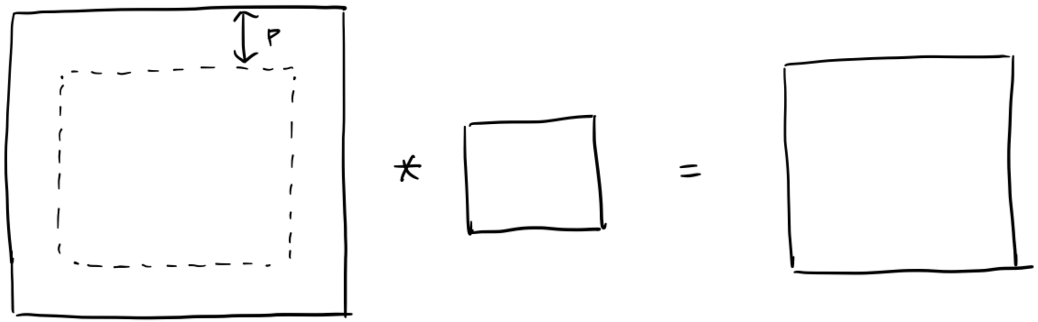
- Add extra zeros around.
- It allows us to use a CONV layer without necessarily shrinking the height and width of the volumes.
- This is important for building deeper networks, since otherwise the height/width would shrink as we go to deeper layers.



Valid Padding vs. Same Padding



$n \times n \quad * \quad f \times f \quad = \quad (n-f+1) \times (n-f+1)$



$(n+2p) \times (n+2p) \quad * \quad f \times f \quad = \quad (n+2p-f+1) \times (n+2p-f+1)$

"Valid" Conv : $p = 0$

"Same" Conv : $p = \frac{f-1}{2}$

Valid and Same Convolutions

→ no padding

"Valid": $n \times n \;\; * \;\; f \times f \;\; \longrightarrow \;\; n-f+1 \times n-f+1$

$6 \times 6 \;\; * \;\; 3 \times 3 \;\; \longrightarrow \;\; 4 \times 4$

"Same": Pad so that output size is the same as the input size.

$n+2p-f+1 \times n+2p-f+1$

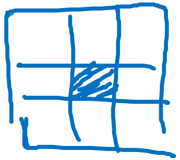
$\cancel{n}+2p-f+1 = \cancel{n} \Rightarrow \boxed{p = \frac{f-1}{2}}$

$3 \times 3 \quad p = \frac{3-1}{2} = 1 \quad \Big| \quad 5 \times 5 \quad p = 2$

$f=5$

f is usually odd

1×1
3×3
5×5
7×7



deeplearning.ai

Convolutional Neural Networks

Strided Convolutions

Strided convolution

2	3	7	4	6	2	9
6	6	9	8	7	4	3
3	4	8	3	8	9	7
7	8	3	6	6	3	4
4	2	1	8	3	4	6
3	2	4	1	9	8	3
0	1	3	9	2	1	4

7×7

*

3	4	4
1	0	2
-1	0	3

3×3

=

91	100	83
69	91	127
44	72	74

3×3

stride = 2

$\lfloor z \rfloor = \text{floor}(z)$

$n \times n$ * $f \times f$

padding p stride s

$s = 2$

$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

$$\frac{7+0-3}{2} + 1 = \frac{4}{2} + 1 = 3$$

Summary of convolutions

$n \times n$ image $\qquad f \times f$ filter

padding $p \qquad$ stride s

Output size:

$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \underbrace{\frac{n+2p-f}{s}} + 1 \right\rfloor$$

Technical note on cross-correlation vs. convolution (Optional)

Convolution in math textbook:

2	3	7	4	6	2
6	6	9	8	7	4
3	4	8	3	8	9
7	8	3	6	6	3
4	2	1	8	3	4
3	2	4	1	9	8

*

3	4	5
1	0	2
-1	9	7

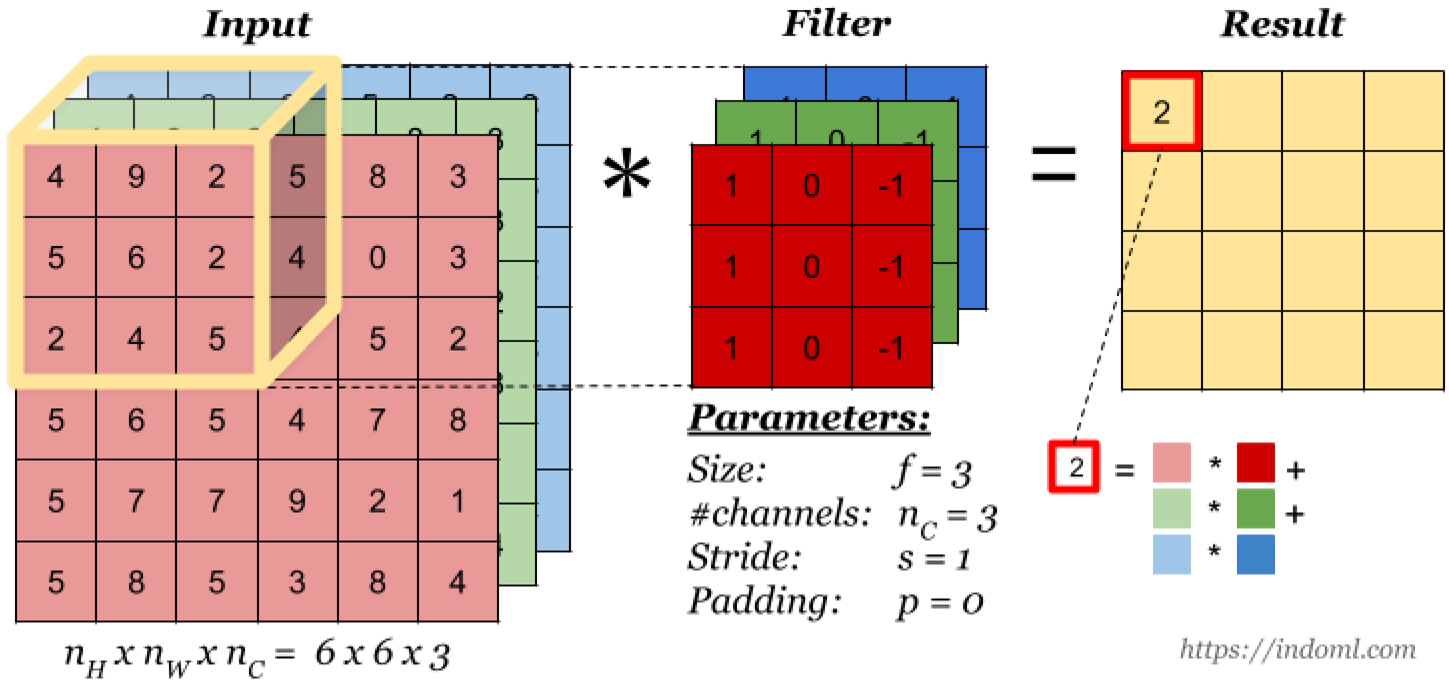
deeplearning.ai

Convolutional Neural Networks

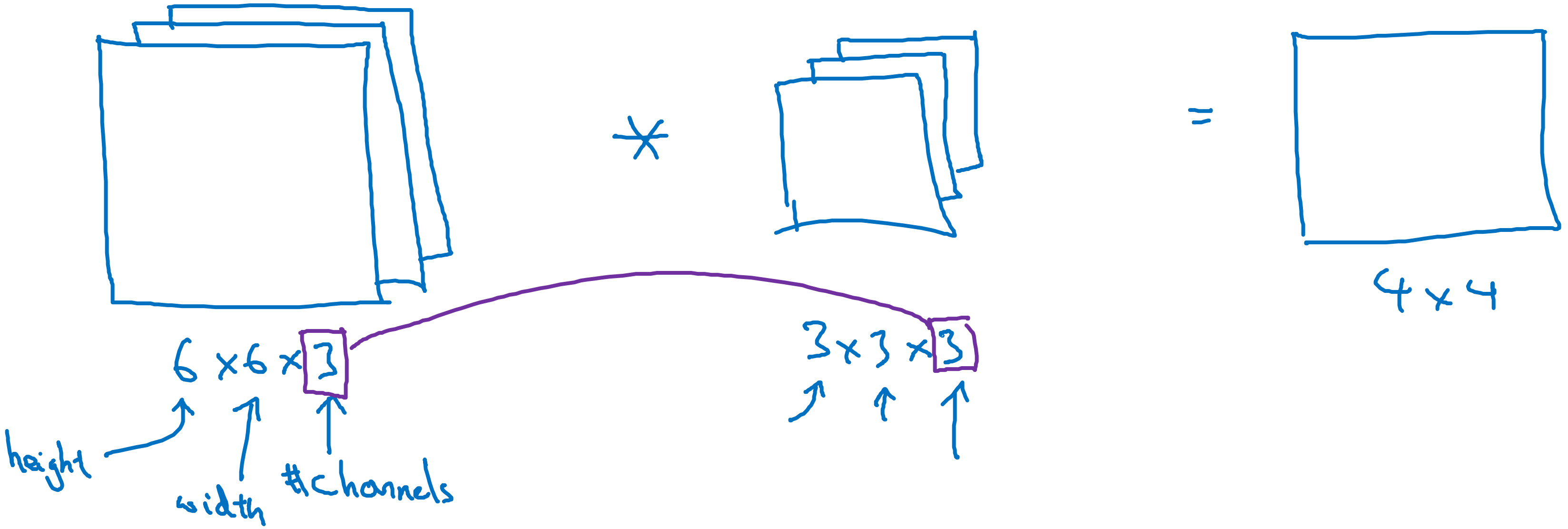
Convolutions over Volumes

Convolution Operation on Volume

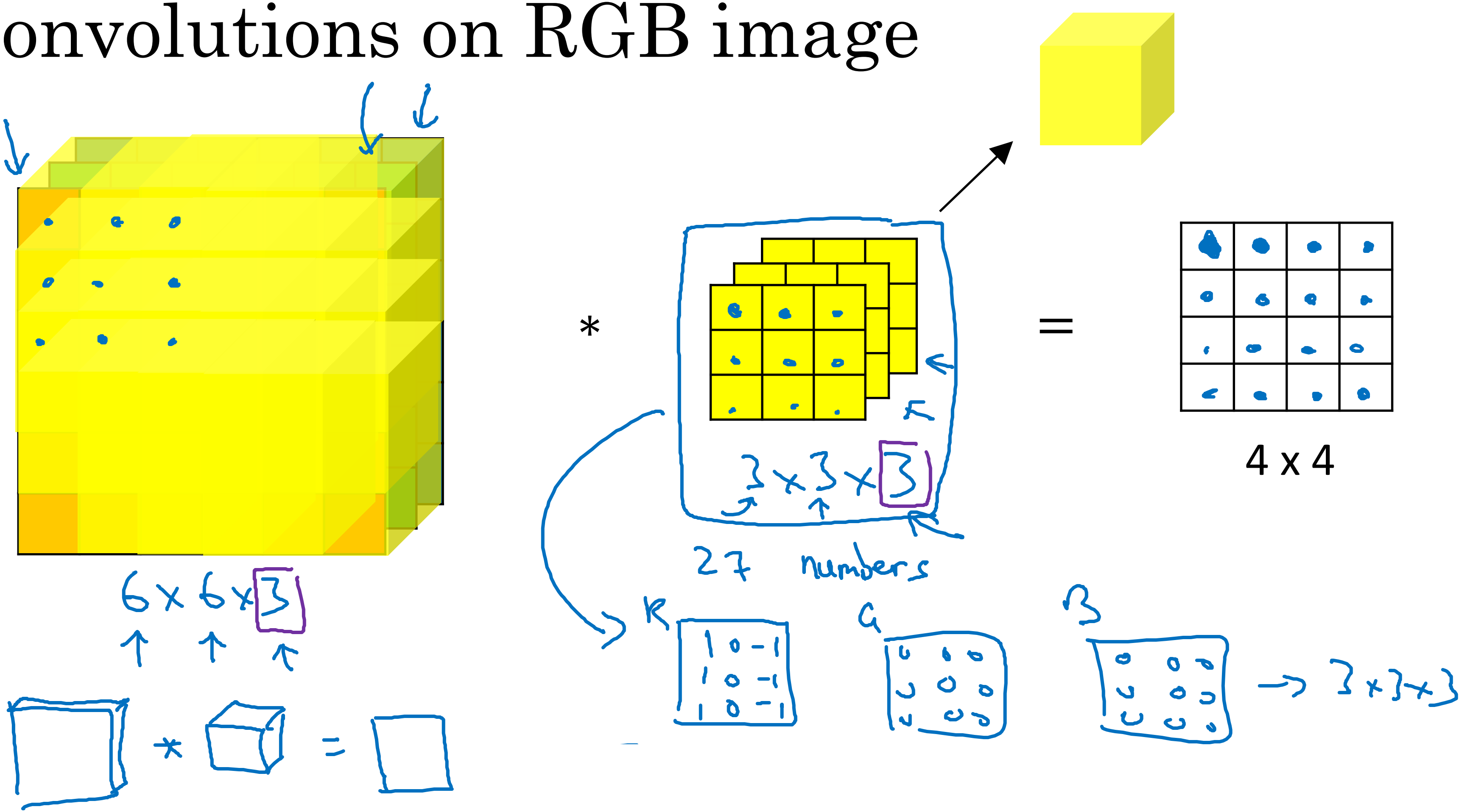
- When input has more than one channels (e.g. an RGB image), the filter should have matching number of channels.



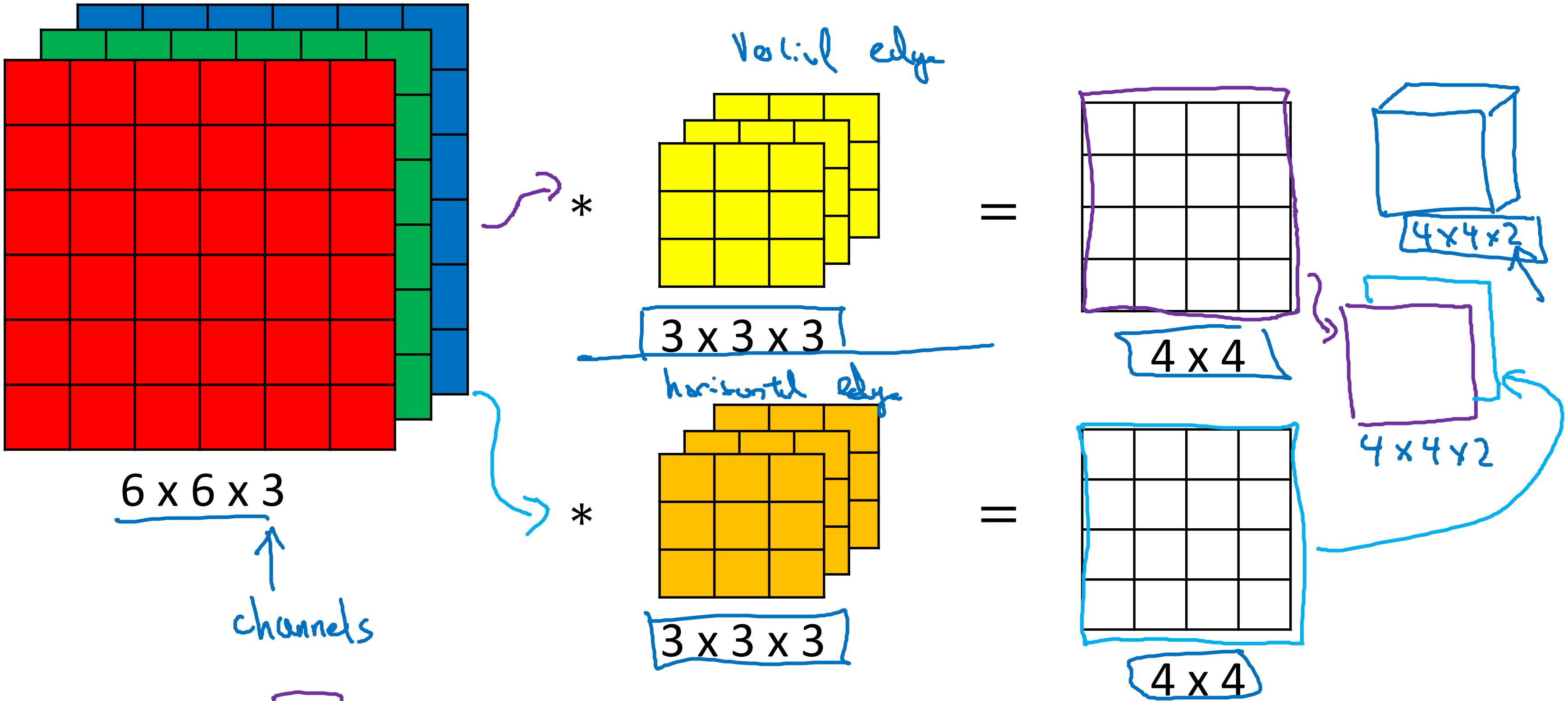
Convolutions on RGB images



Convolutions on RGB image



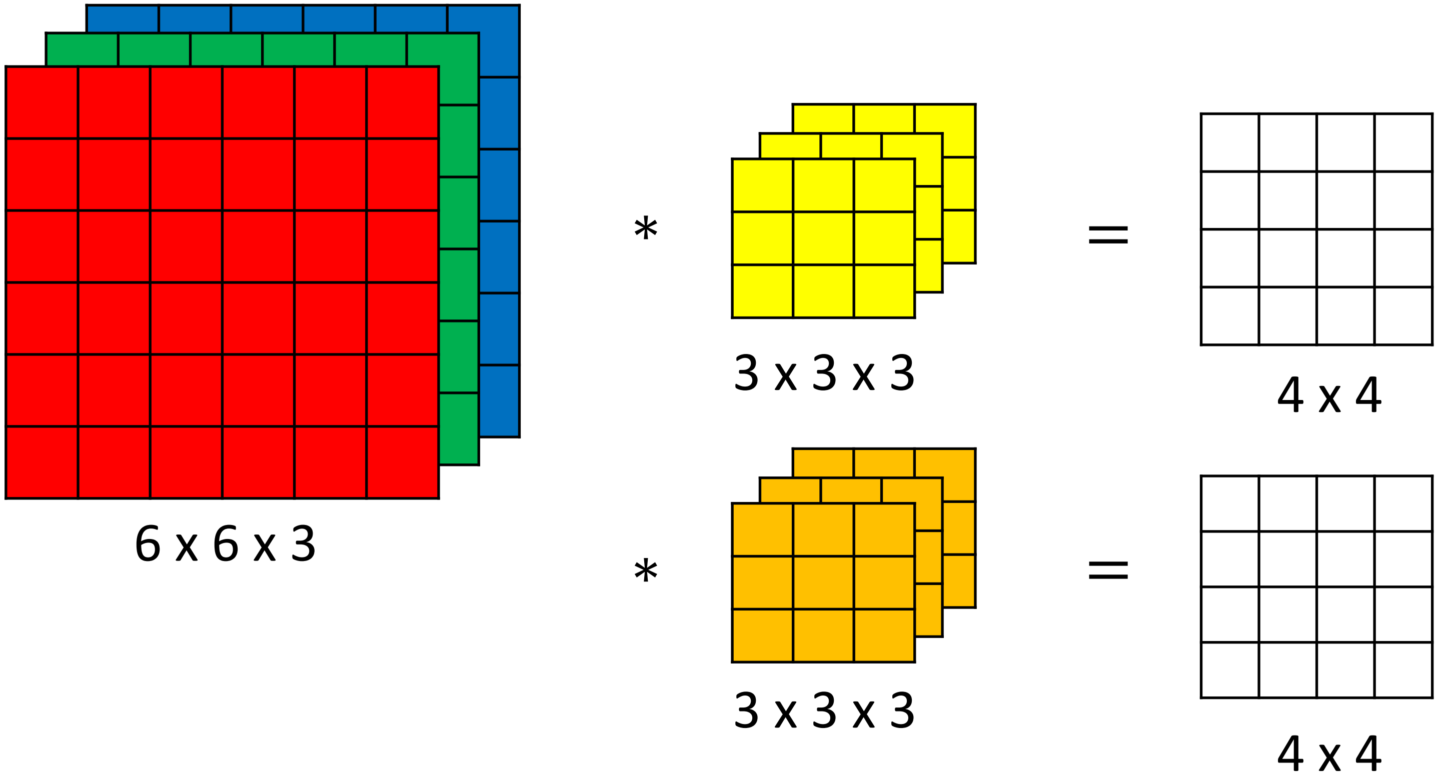
Multiple filters



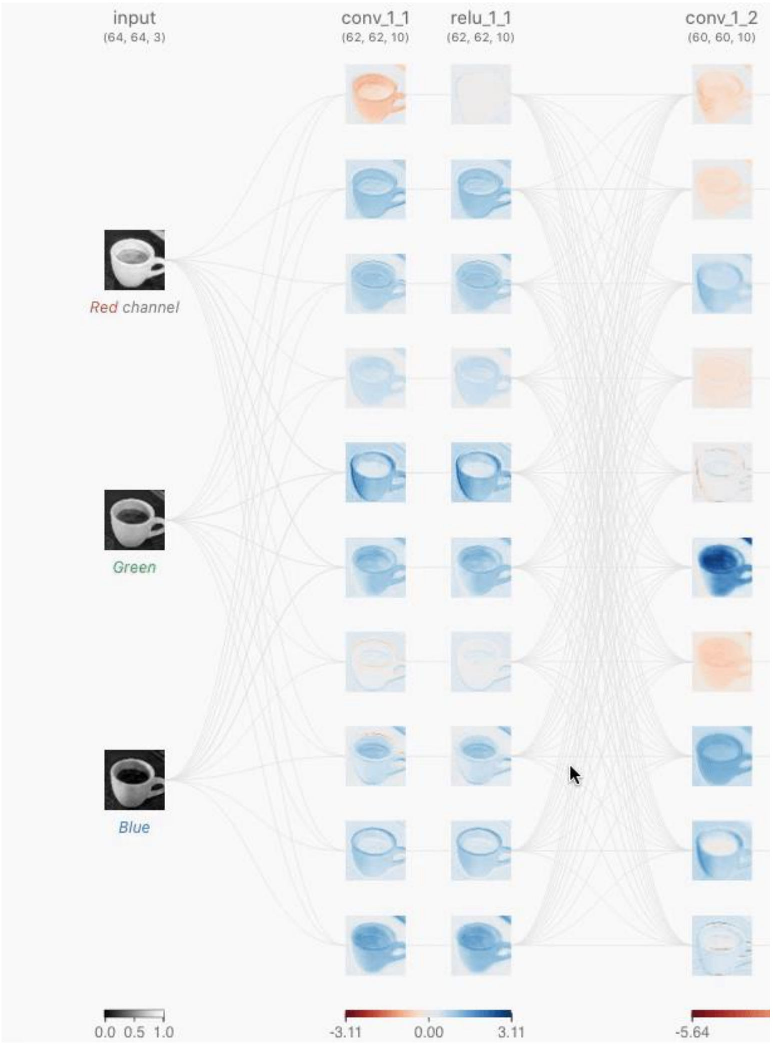
Summary: $n \times n \times \boxed{n_c}$ $*$ $f \times f \times \boxed{n_c}$ $\longrightarrow$ $\underbrace{n-f+1}_{4} \times \underbrace{n-f+1}_{4} \times \underbrace{n_c'}_{2}$ ← # filters

$6 \times 6 \times 3$ $\quad$ $3 \times 3 \times 3$

Example of a layer (with bias & activation function)



Example:



input
(64, 64, 3)
conv_1_1
(62, 62, 10)
relu_1_1
(62, 62, 10)
conv_1_2
(60, 60, 10)
Red channel
Green
Blue
0.0 0.5 1.0
-3.11
0.00
3.11
-5.64

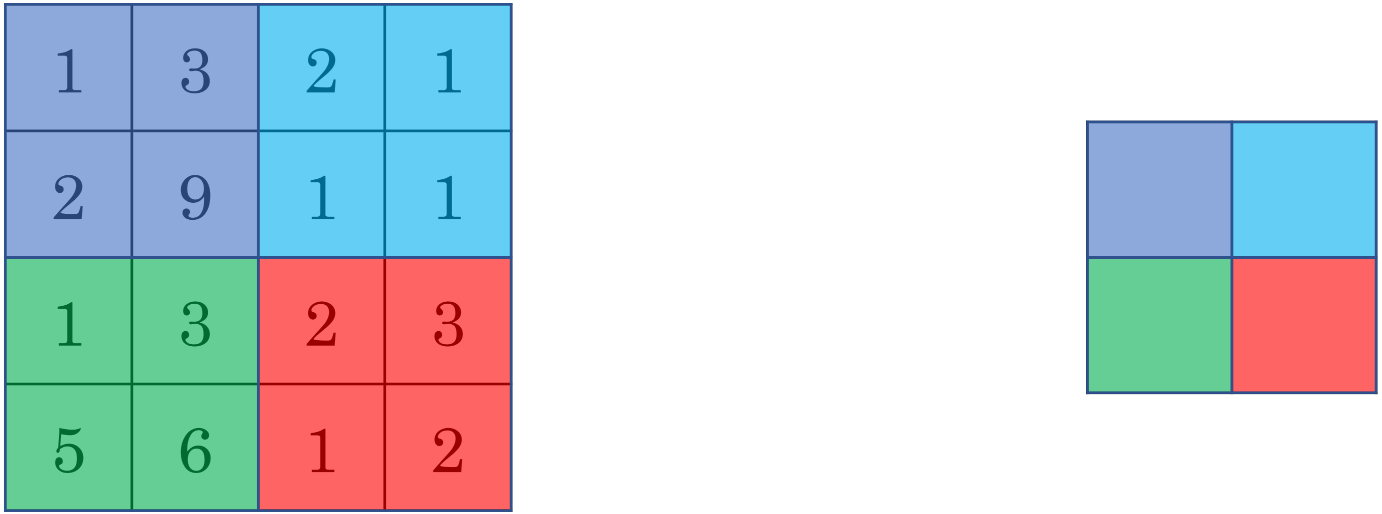
deeplearning.ai

Convolutional Neural Networks

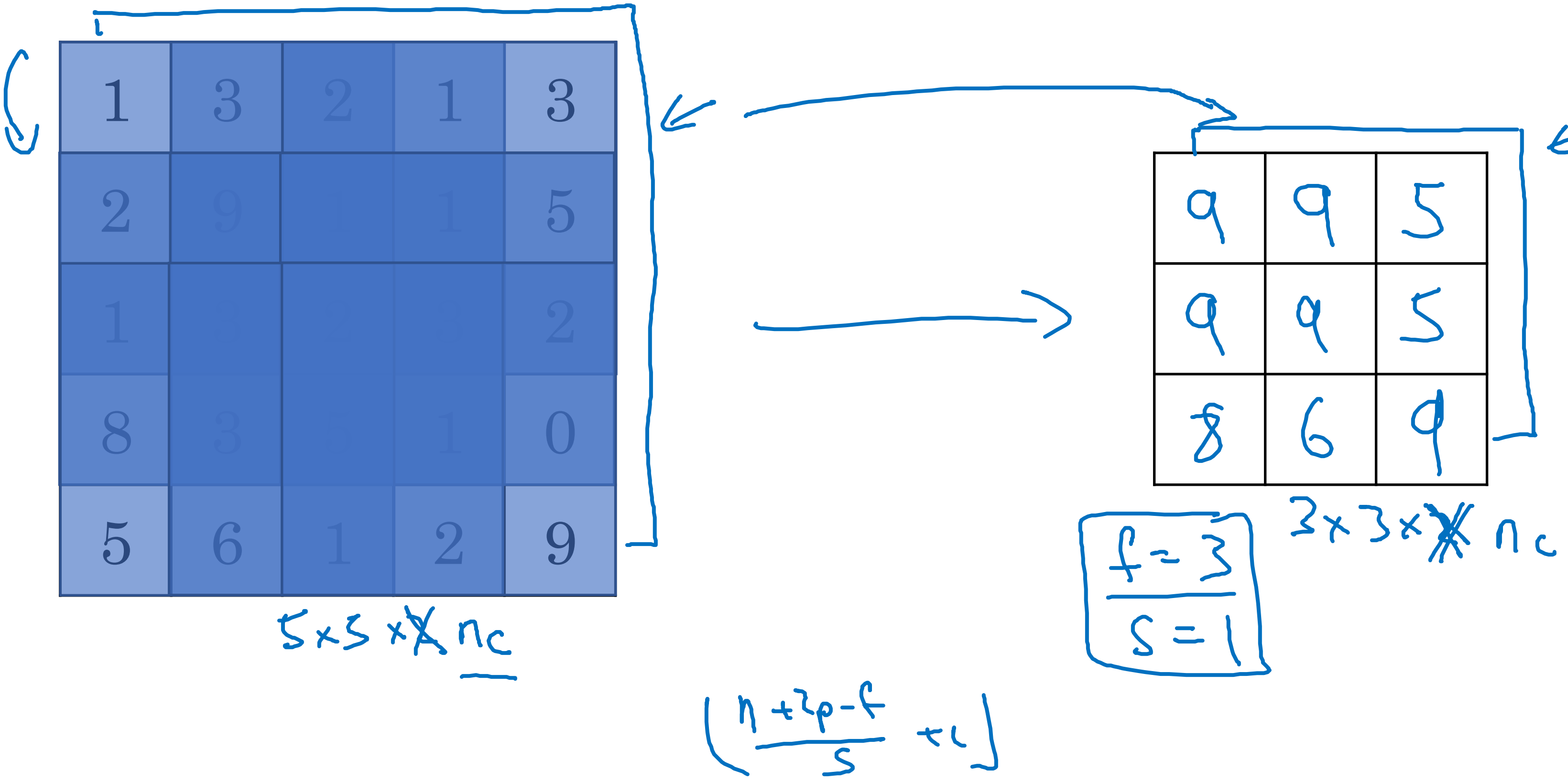
Pooling layers

Pooling layer: Max pooling

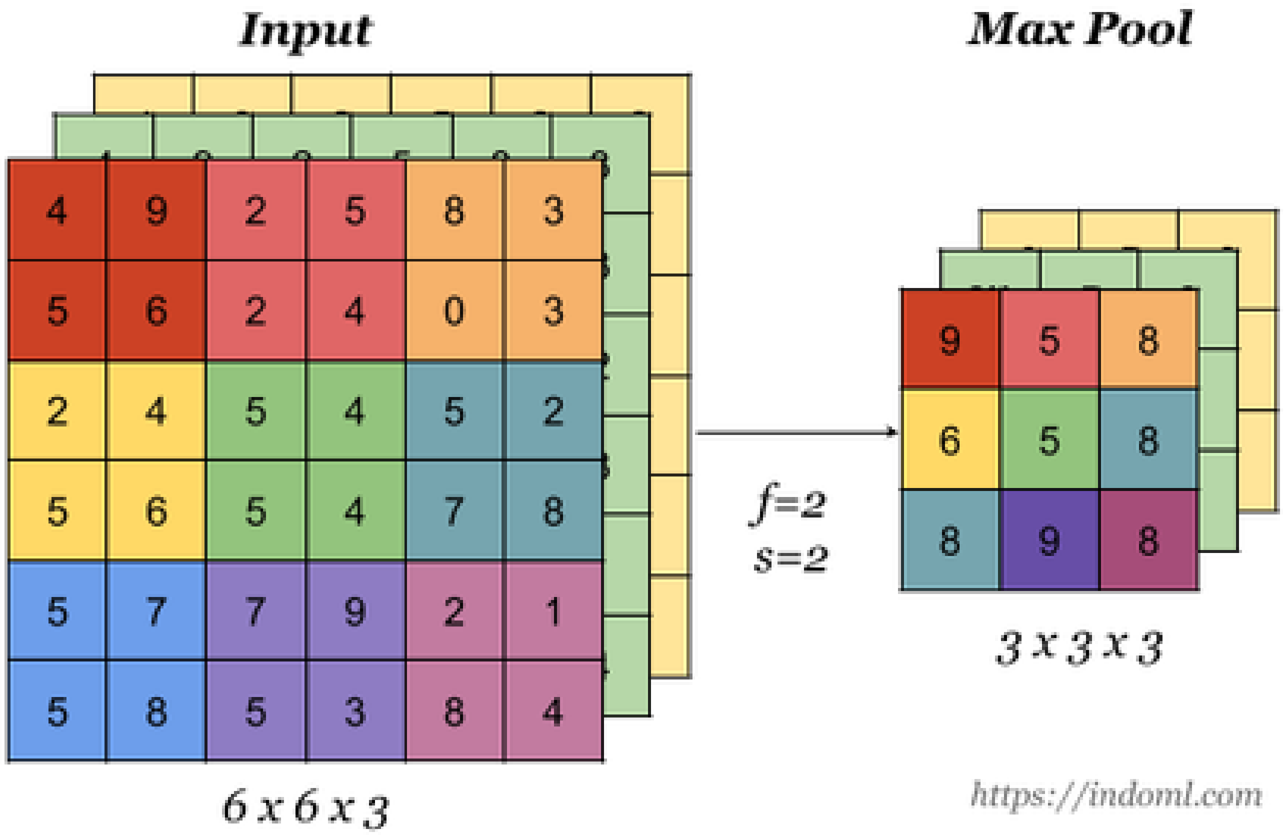
Pool layer reduces the size of the inputs to speed up computation and make features more robust.



Pooling layer: Max pooling



Max Pooling with multiple Channels



Pooling layer: Average pooling

1	3	2	1
2	9	1	1
1	4	2	3
5	6	1	2

→

3.75	1.25
4	2

$f=2$

$s=2$

$\underline{7 \times 7} \times 1000 \rightarrow 1 \times 1 \times 1000$

Summary of pooling

Hyperparameters:

f : filter size

s : stride

type: Max or Average pooling

$f=2, s=2$

$f=3, s=2$

~~p: padding.~~

No parameters to learn!

$n_H \times n_W \times n_C$

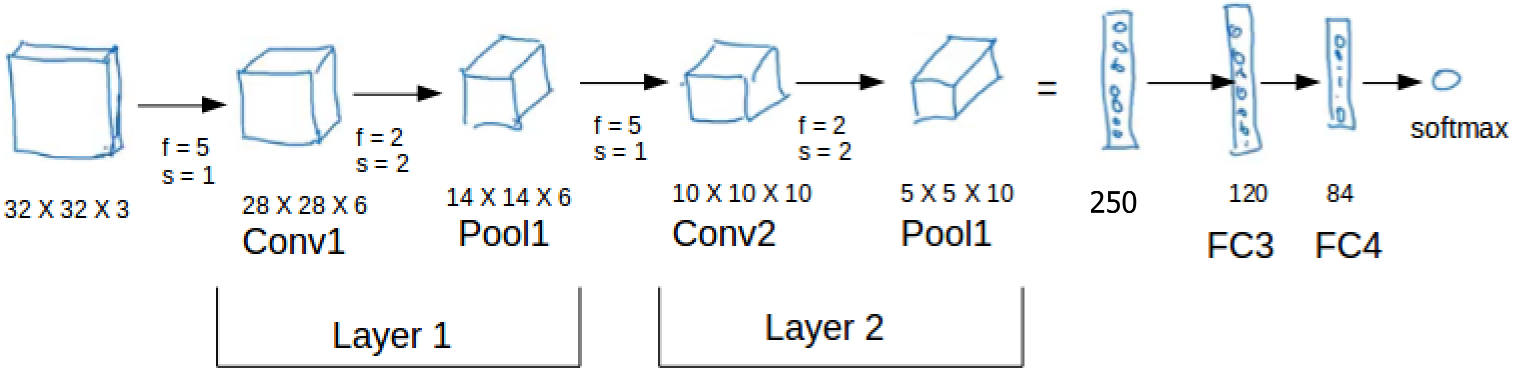
$\downarrow$

$\left\lfloor \frac{n_H - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_W - f}{s} + 1 \right\rfloor \times n_C$

there's nothing for gradient descent to learn!

Simple CNN Example

- In most Conv networks, as we propagate forward, the filter sizes get bigger and the outputs get smaller.
- Towards the end, for classification purposes, we unfold (flattening) all the features to use Fully Connected (FC) layers.
  - Fully connected layer involves weights, biases, and neurons. It connects neurons in one layer to neurons in another layer.
- Finally a Softmax Layer to classify the input into various categories.



Softmax

- The **softmax function,** also known as **softargmax** or **normalized exponential function**, is a generalization of the logistic function to multiple dimensions.
- Takes as input a vector ***z*** of K real numbers, and normalizes it into a probability distribution consisting of *K* probabilities proportional to the exponentials of the input numbers.
- Prior to applying softmax, some vector components could be negative, or greater than one; and might not sum to 1; but after applying softmax, each component will be in the interval (0,1), and the components will add up to 1.

$$e(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

$$\begin{bmatrix} 1.2 \\ 0.9 \\ 0.4 \end{bmatrix} \longrightarrow \boxed{\text{Softmax}} \longrightarrow \begin{bmatrix} 0.46 \\ 0.34 \\ 0.20 \end{bmatrix}$$

$$z = (z_1, z_2, .., z_K) \in R^K$$

Softmax assumes that each example is a member of exactly one class. Some examples, however, can simultaneously be a member of multiple classes. For such examples:

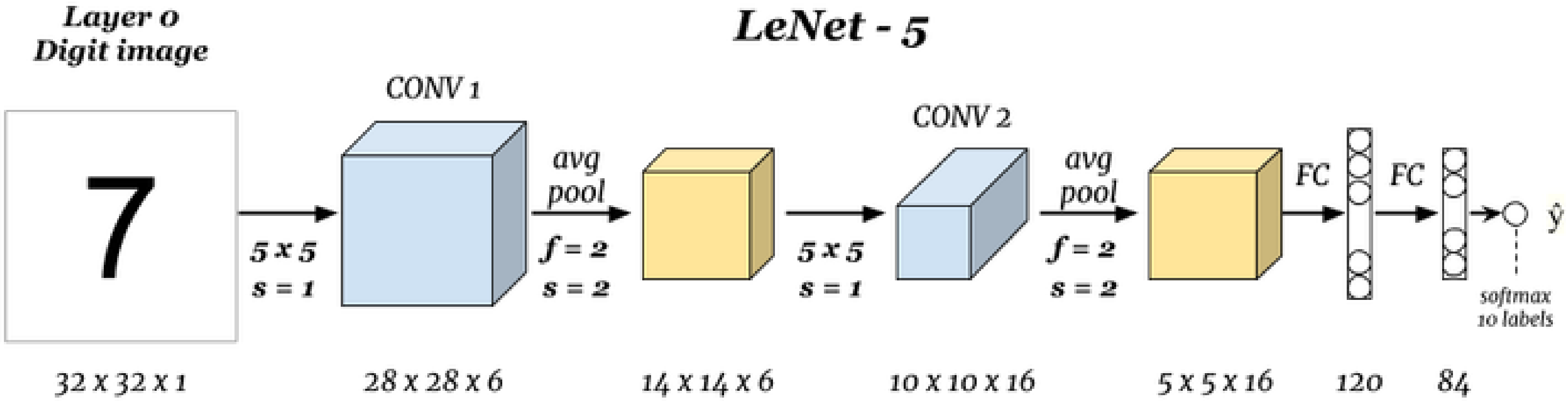
- You may not use Softmax.
- You must rely on multiple logistic regressions.

deeplearning.ai

Convolutional Neural Networks

Well Known Architectures

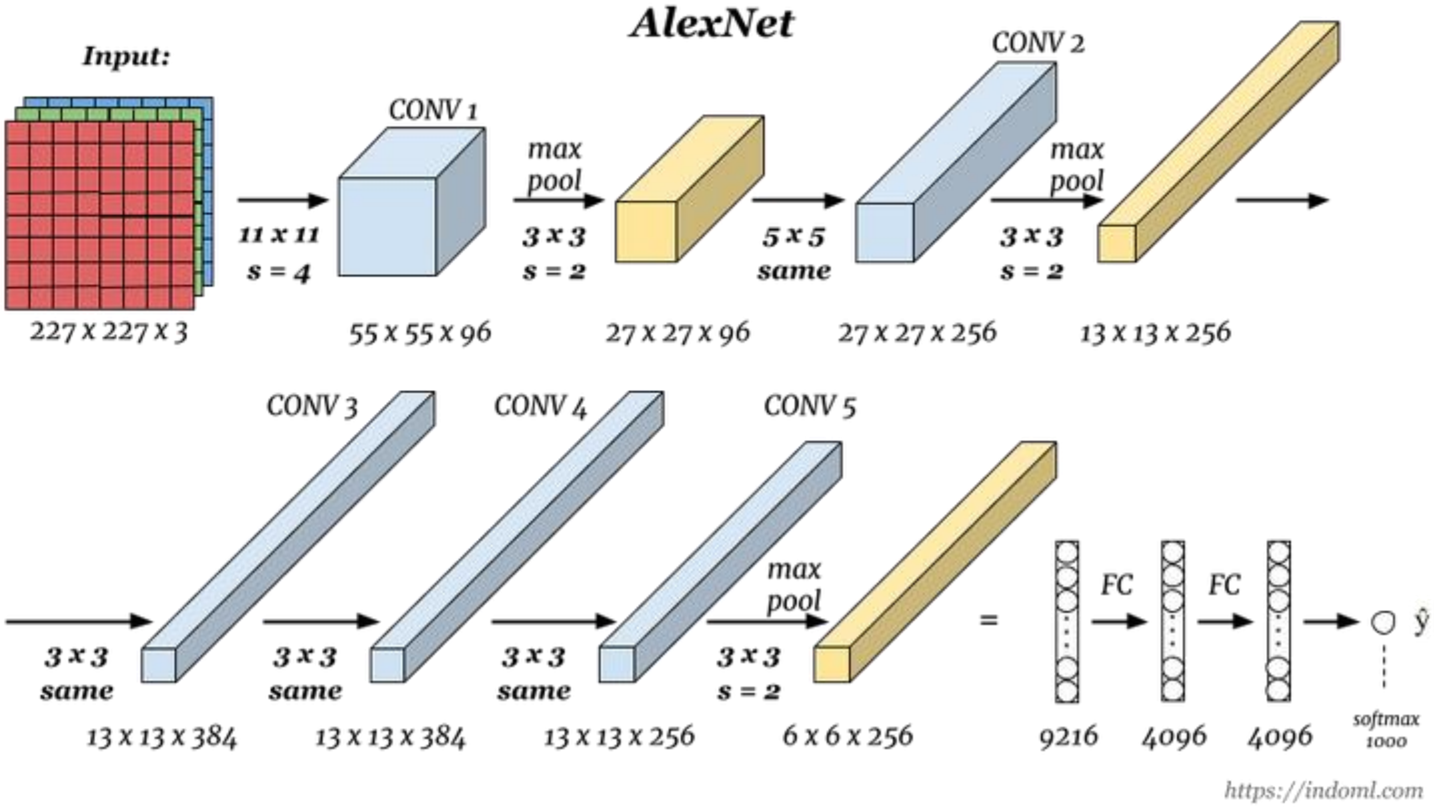
LeNet - 5



- Number of parameters: ~ 60 thousands.

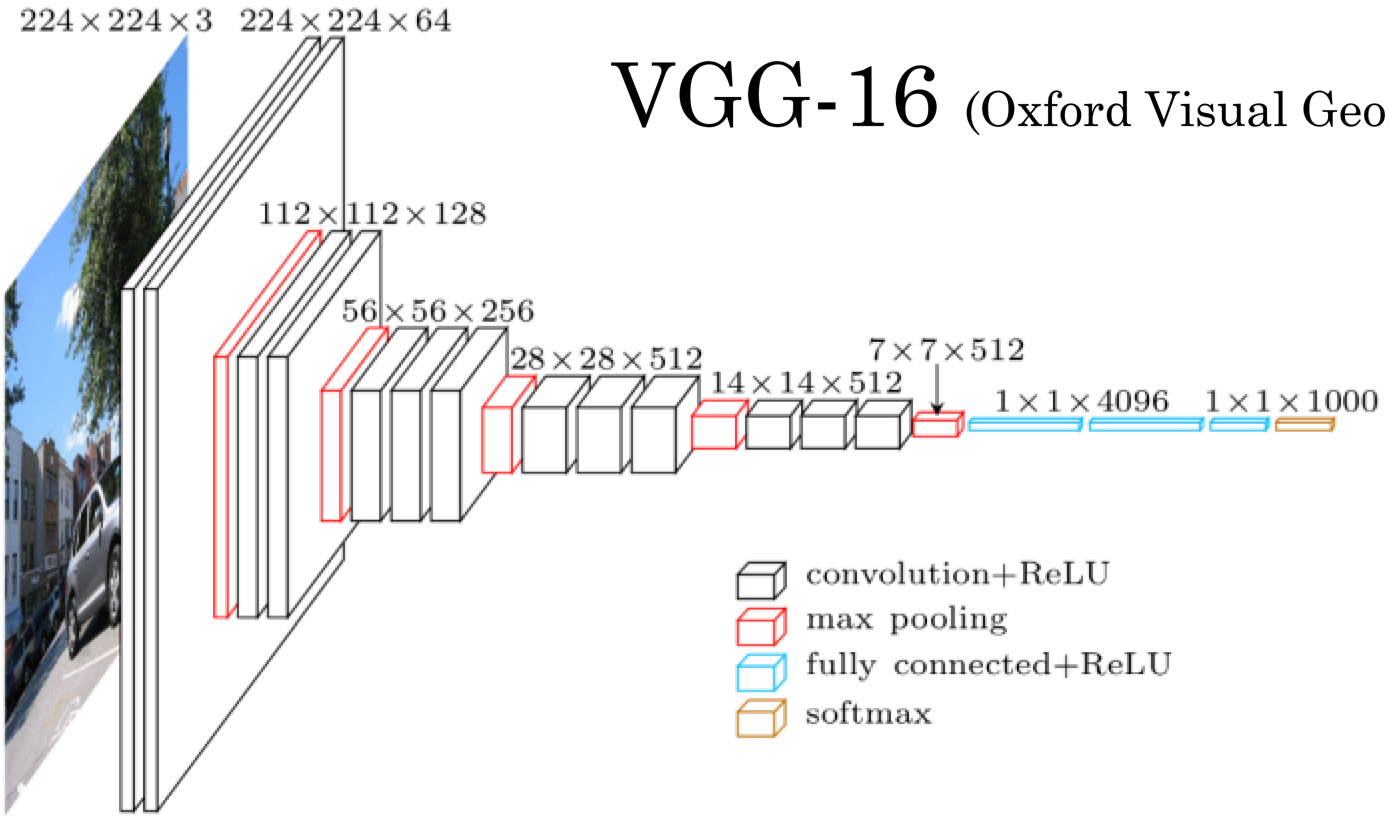
Gradient-Based Learning Applied to Document Recognition paper by Y. Lecun, L. Bottou, Y. Bengio and P. Haffner (1998)

AlexNet



- Similar to LeNet-5 with just more convolution and pooling layers:
- Number of parameters: ~ 60 million.

[ImageNet Classification with Deep Convolutional Neural Networks](#) paper by Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever (2012).



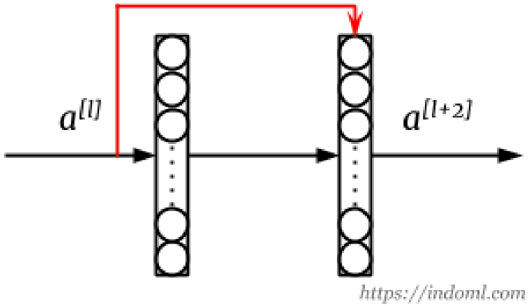
VGG-16 (Oxford Visual Geometry Group)

- Number of parameters: ~ 138 millions.
- The strength is in the simplicity: the dimension is halved and the depth is increased on every step (or stack of layers)

Very Deep Convolutional Networks for Large-Scale Image Recognition paper by Karen Simonyan and Andrew Zisserman (2014).

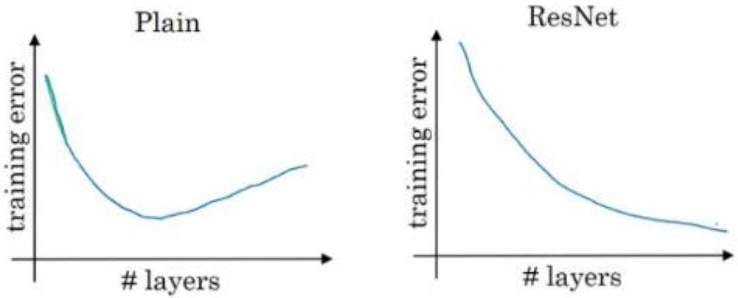
ResNet

- The problem with deeper neural networks are they are harder to train and once the number of layers reach certain number, the training error starts to raise again.
- Deep networks are also harder to train due to ***exploding*** and ***vanishing*** gradients problem.
- Residual Network solves these problems by implementing skip connection where output from one layer is fed to layer deeper in the network



$$z^{[l+2]} = W^{[l+2]} a^{[l+1]} + b^{[l+2]}$$

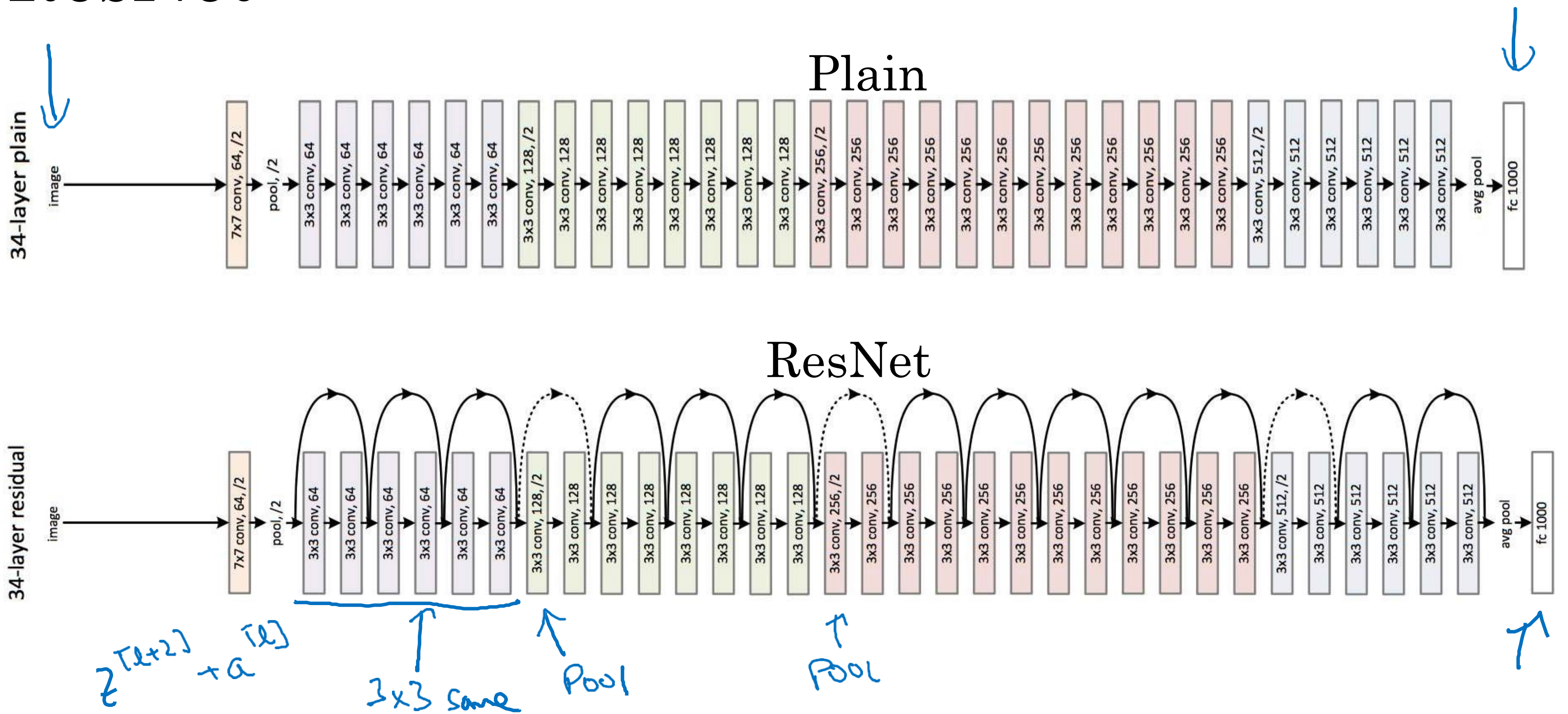
$$a^{[l+2]} = g^{[l+2]}(z^{[l+2]} + a^{[l]})$$



The benefit of training a residual network is that even if we train deeper networks, the training error does not increase.

He et al. in [Deep Residual Learning for Image Recognition paper](#) (2015)

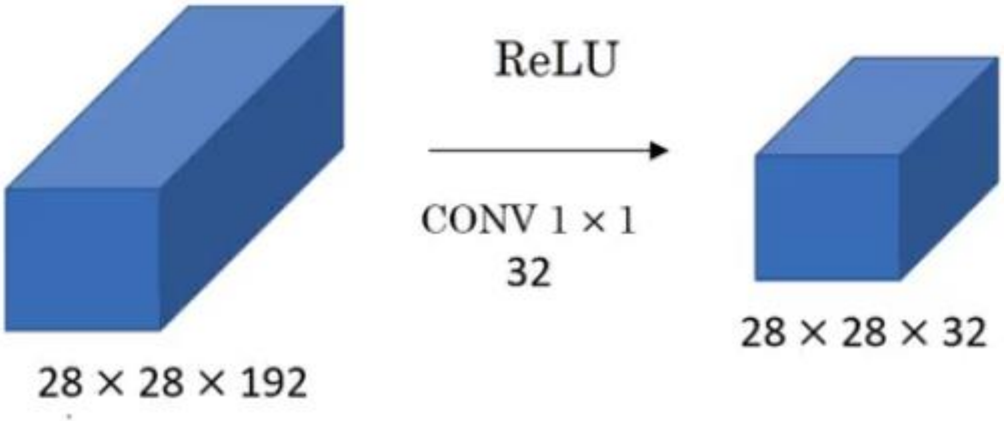
ResNet



He et al. in *Deep Residual Learning for Image Recognition paper* (2015)

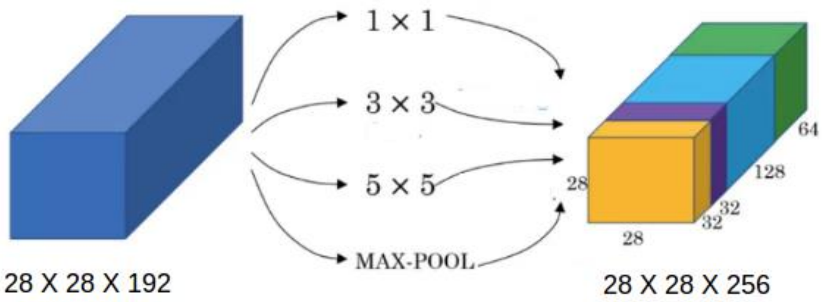
1×1 Convolutions

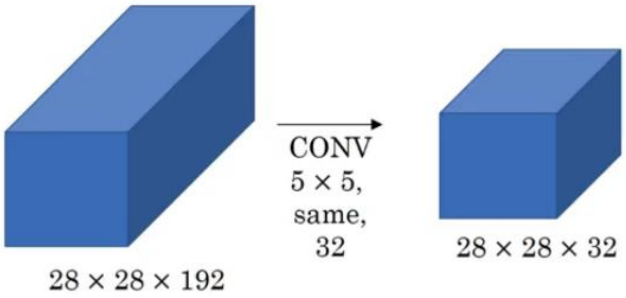
- The basic idea of using 1 X 1 convolution is to reduce the number of channels from the image.
  - We generally use a pooling layer to shrink the height and width of the image
  - To reduce the number of channels from an image, we convolve it using a 1 X 1 filter (hence reducing the computation cost as well)



Inception Network - Motivation

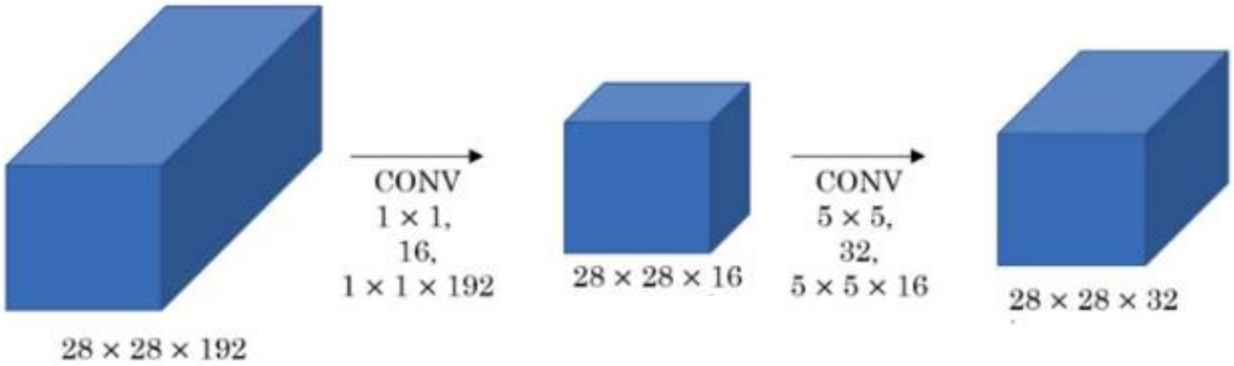
- The motivation of the inception network is, rather than requiring us to pick the filter size manually, let the network decide what is best to put in a layer.
- We give it choices and hopefully it will pick up what is best to use in that layer:



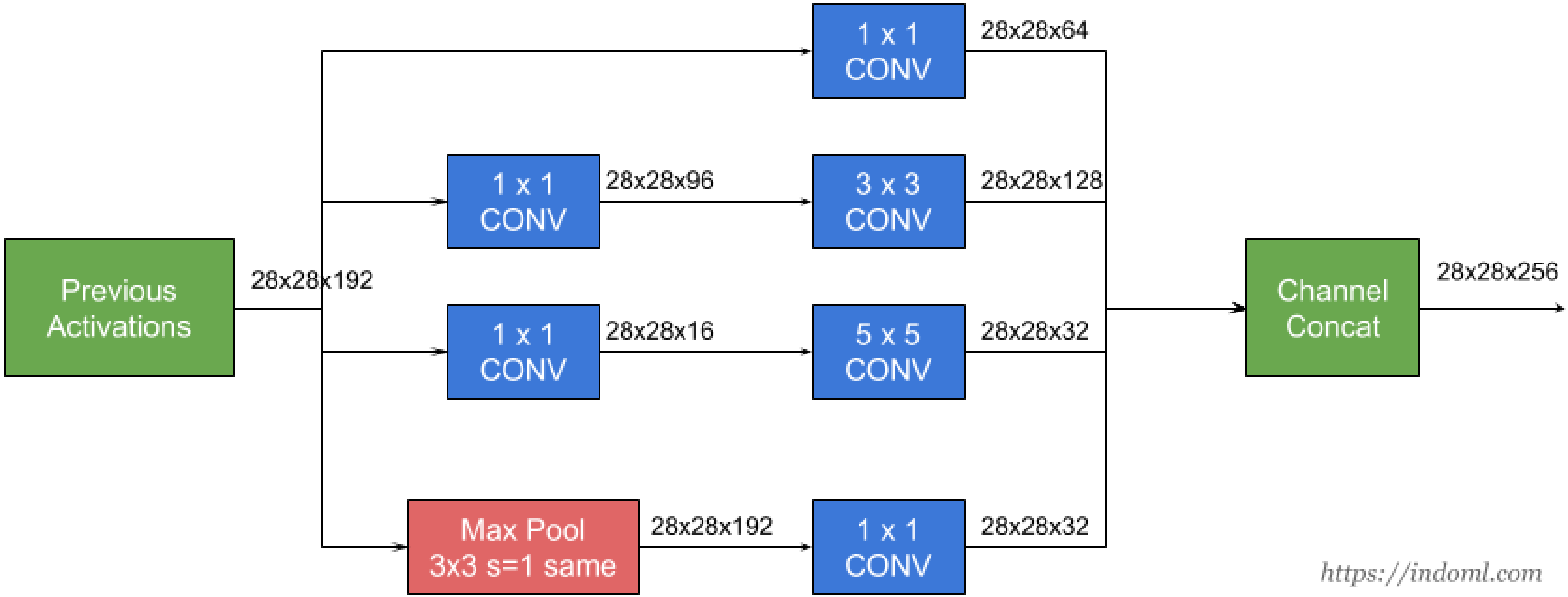


Inception Network - Motivation

- Let’s look at the computations a 1 X 1 convolution and then a 5 X 5 convolution will give us:



Inception Module



Inception Network (V1)

- Inception network called **GoogLeNet**, described in *Going Deeper with Convolutions paper* by Szegedy et al. (2014), (Winner ILSCVC 2014)
  - has 9 inception modules