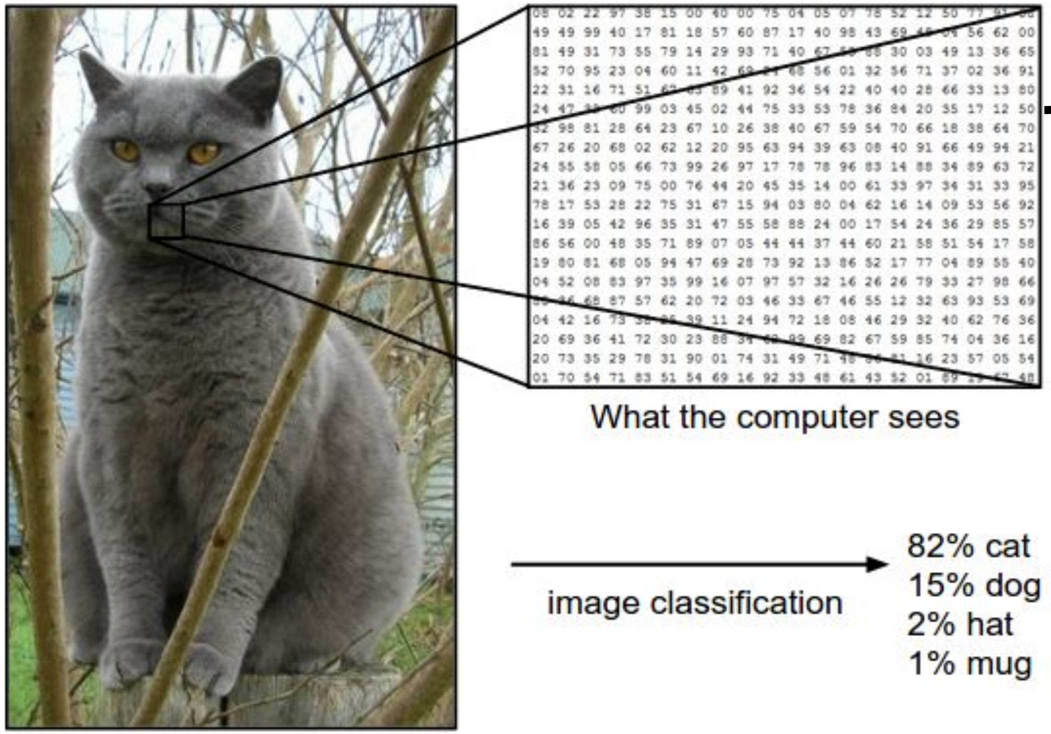
CNN BASICS

Convolutional Neural Networks have been some of the most influential innovations in the field of computer vision

What we see ?

What computer gets ?

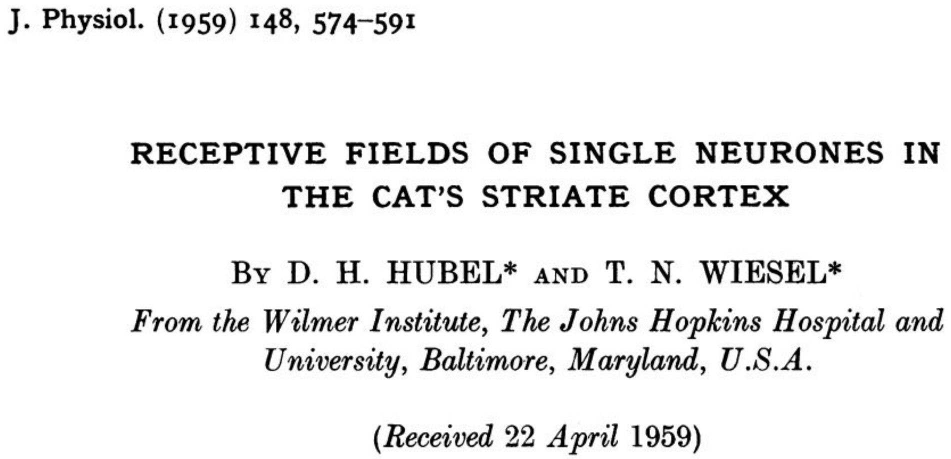


Images are 3-dimensional arrays of integers from 0 to 255, of size Width x Height x 3. The 3 represents the three color channels Red, Green, Blue.

http://cs231n.github.io/classification/

What CNNs do would be that you take the image, pass it through a series of convolutional, nonlinear, pooling (downsampling), and fully connected layers, and get an output.

1959 : David H. Hubel and Torsten Wiesel - Simple and Complex Cell



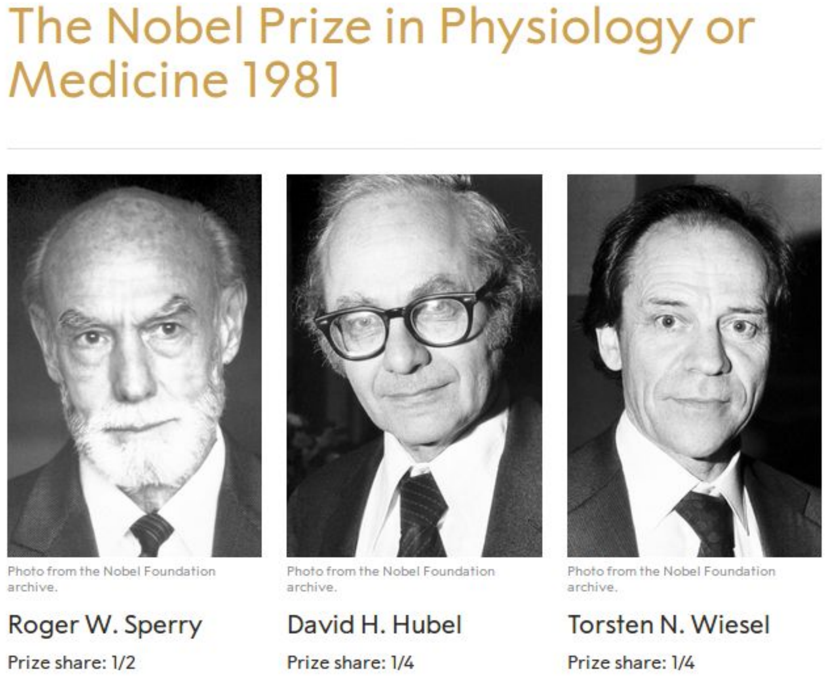
J. Physiol. (1959) 148, 574–591

RECEPTIVE FIELDS OF SINGLE NEURONES IN THE CAT'S STRIATE CORTEX

BY D. H. HUBEL* AND T. N. WIESEL*

From the Wilmer Institute, The Johns Hopkins Hospital and University, Baltimore, Maryland, U.S.A.

(*Received 22 April 1959*)



The Nobel Prize in Physiology or Medicine 1981

Photo from the Nobel Foundation archive.

Roger W. Sperry

Prize share: 1/2

Photo from the Nobel Foundation archive.

David H. Hubel

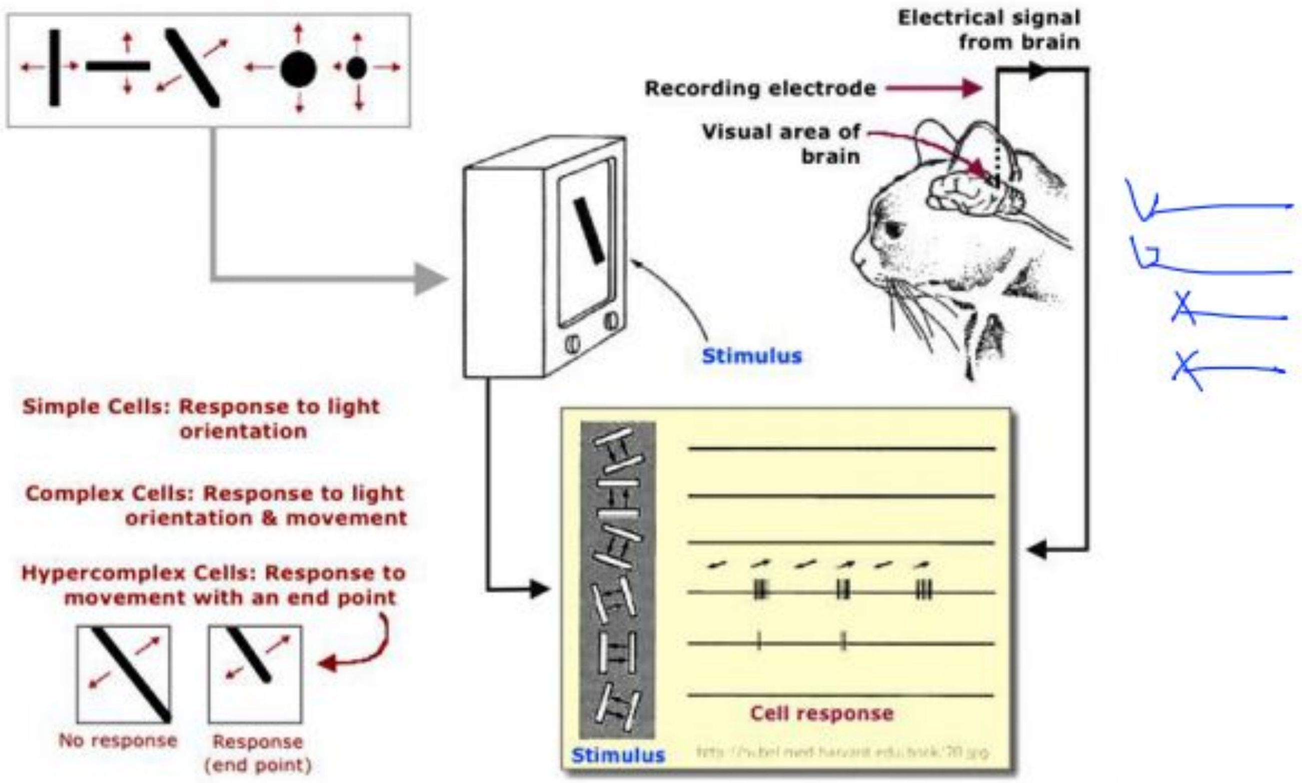
Prize share: 1/4

Photo from the Nobel Foundation archive.

Torsten N. Wiesel

Prize share: 1/4

https://www.nobelprize.org/prizes/medicine/1981/summary/


Electrical signal from brain
Recording electrode
Visual area of brain
Stimulus
Simple Cells: Response to light orientation
Complex Cells: Response to light orientation & movement
Hypercomplex Cells: Response to movement with an end point
No response
Response (end point)
Cell response
Stimulus
V
V
X
X

Hubel & Wiesel, 1959

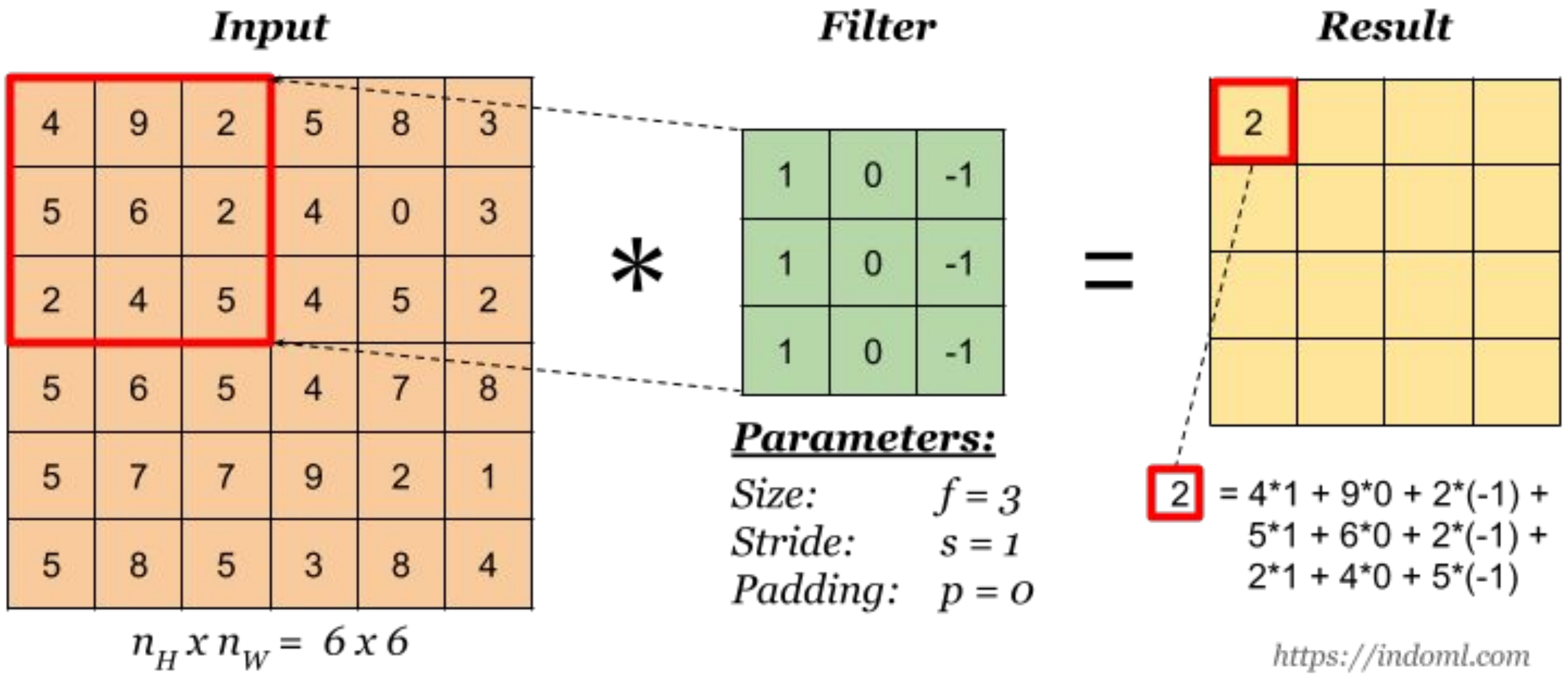
https://korea7030.github.io/Study8/

Why Convolution ?

- **Parameter sharing**: a feature detector (such as a vertical edge detector) that’s useful in one part of the image is probably useful in another part of the image.

- **Sparsity of connections**: in each layer, each output value depends only on small number of inputs.

Basic Convolution Operation



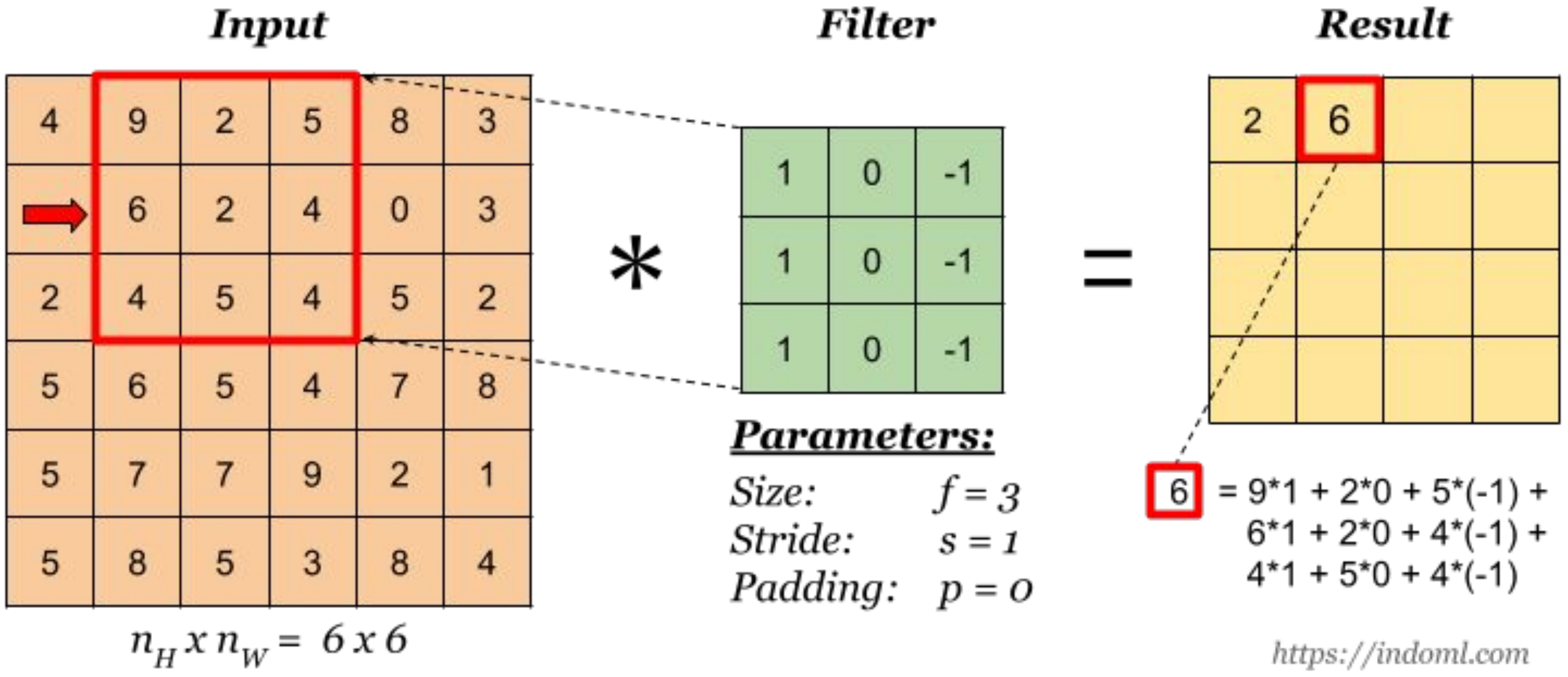
Terminology

Filter size : F

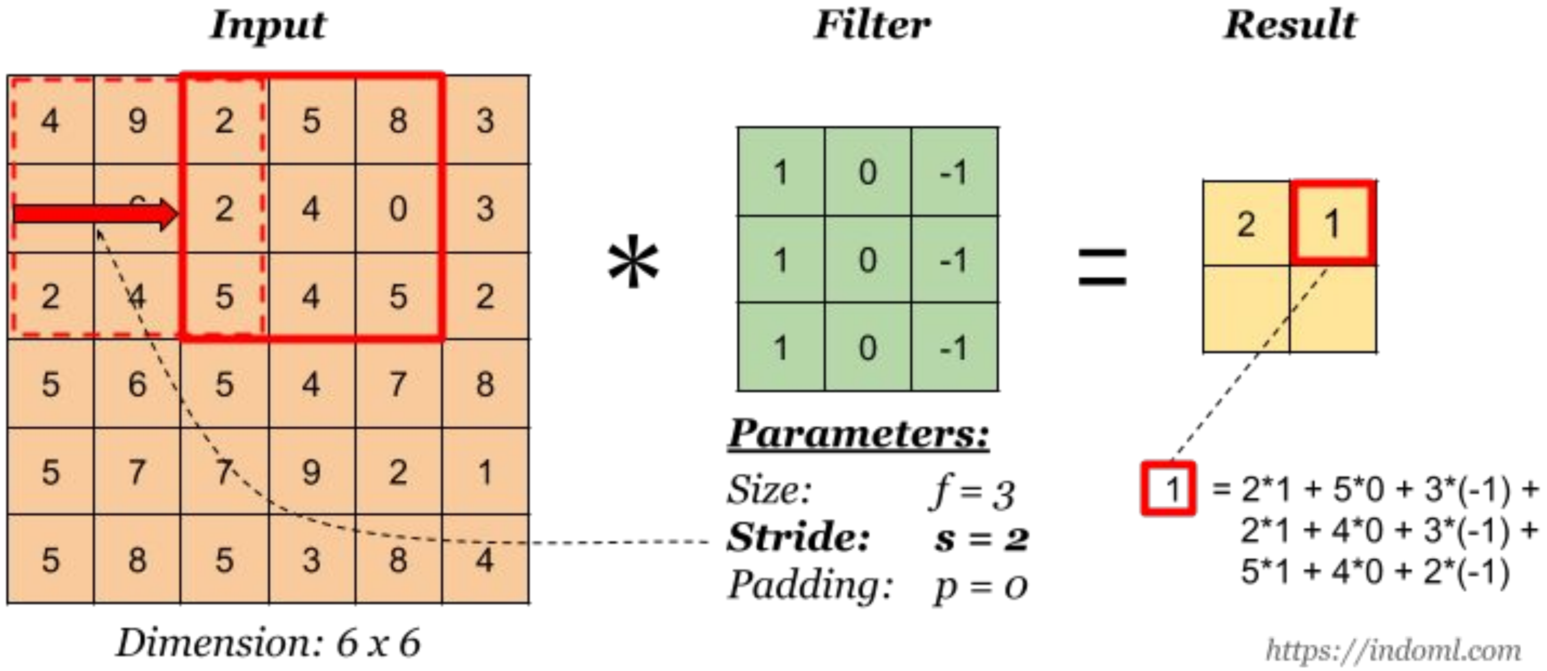
Stride : S

Padding : p

Complete


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
=
Result
2 6
Parameters:
Size: $f = 3$
Stride: $s = 1$
Padding: $p = 0$
6 = 9*1 + 2*0 + 5*(-1) + 6*1 + 2*0 + 4*(-1) + 4*1 + 5*0 + 4*(-1)
https://indoml.com

Stride : Stride governs how many cells the filter is moved in the input to calculate the next cell in the result.



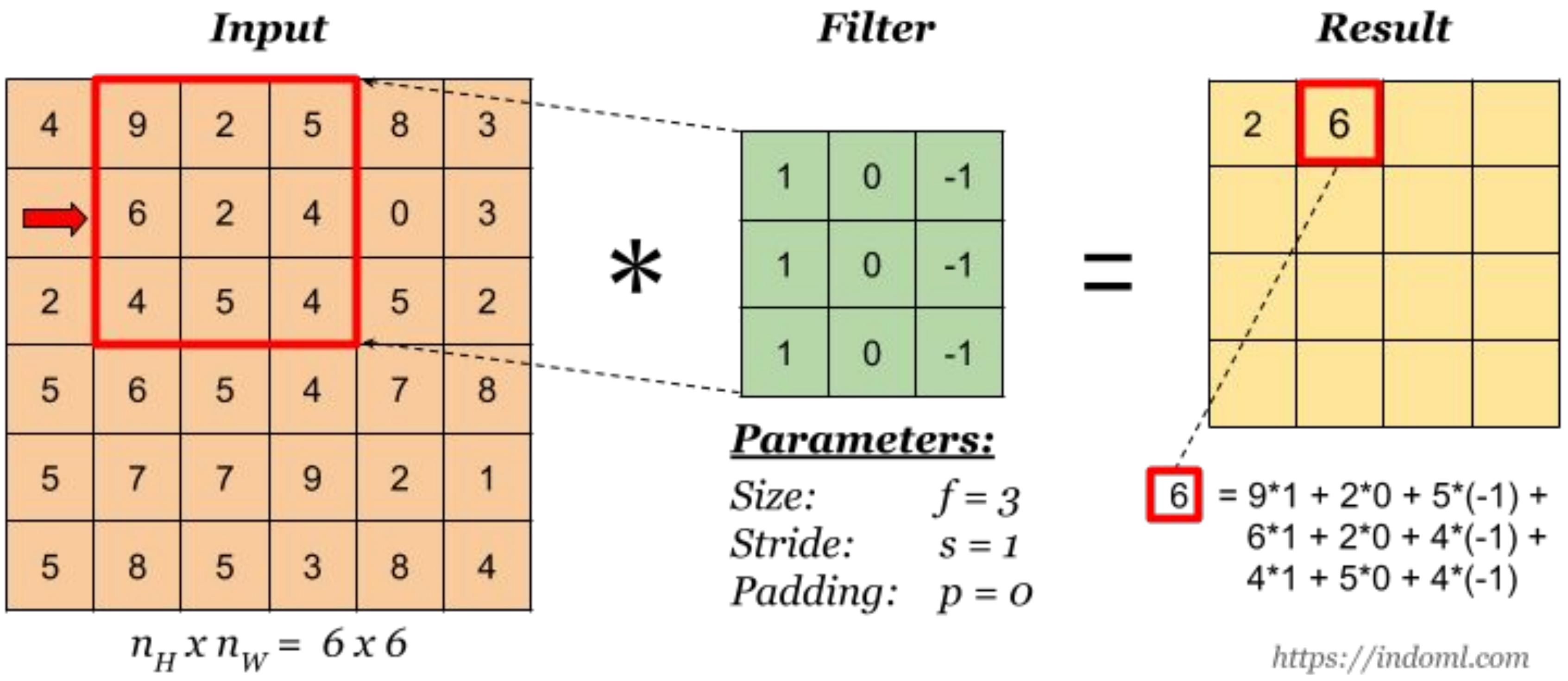
Padding

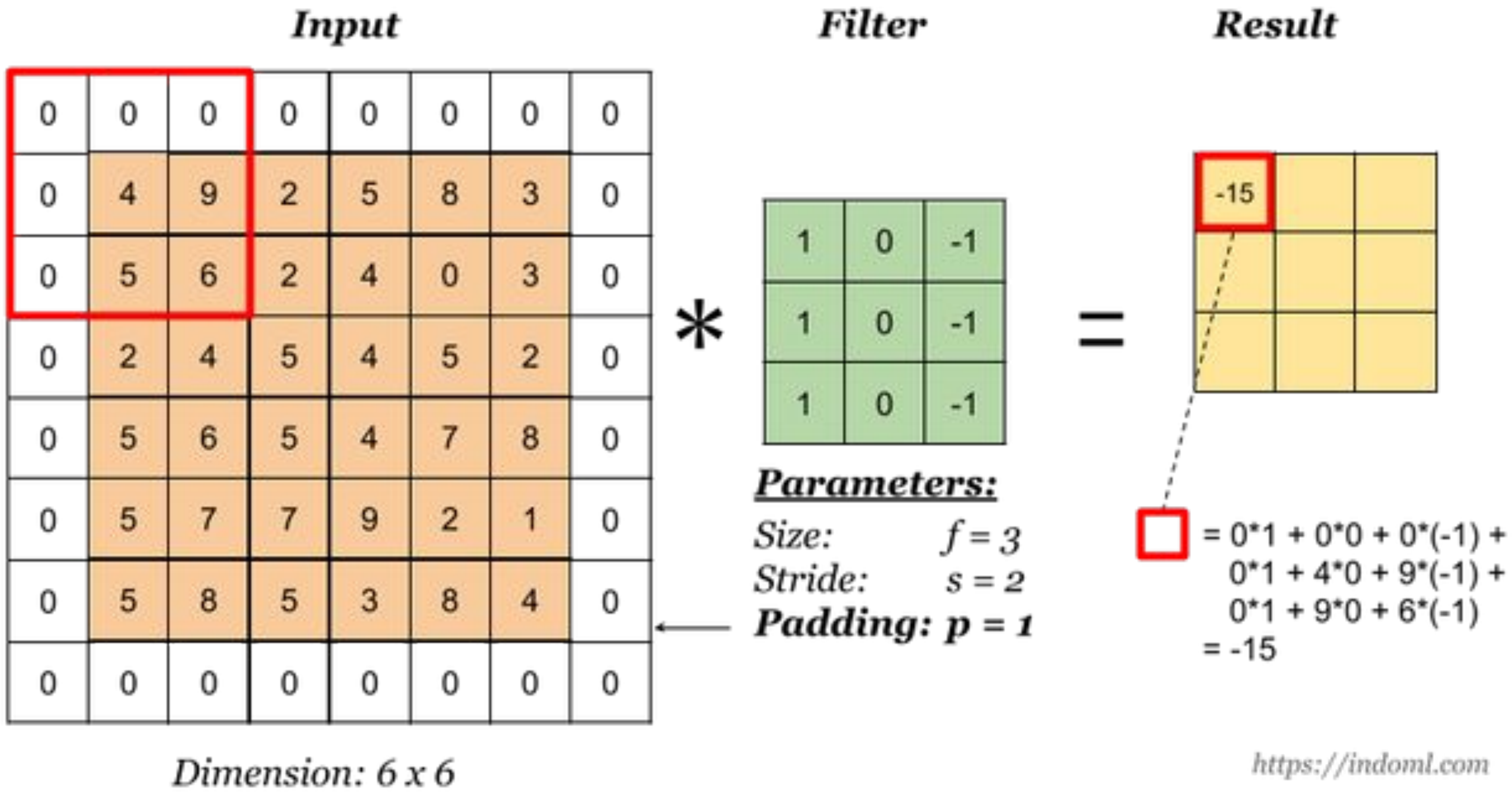
1. It allows us to use a CONV layer without necessarily shrinking the height and width of the volumes. This is important for building deeper networks, since otherwise the height/width would shrink as we go to deeper layers.

2. It helps us keep more of the information at the border of an image. Without padding, very few values at the next layer would be affected by pixels as the edges of an image.

Some padding terminologies:

- “valid” padding: no padding
- “same” padding: padding so that the output dimension is the same as the input

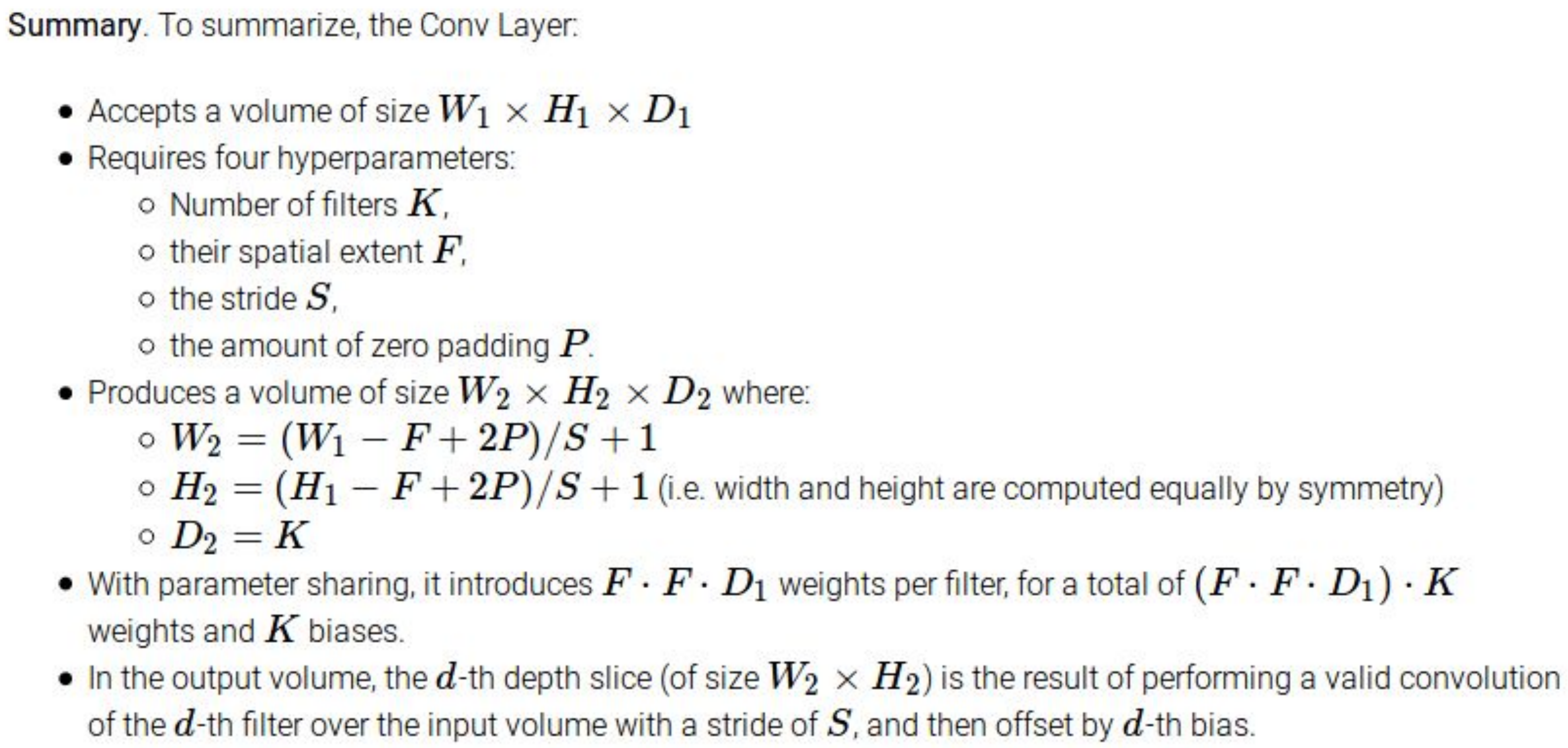

Input
4 9 2 5 8 3
6 2 4 0 3
2 4 5 4 5 2
5 6 5 4 7 8
5 7 7 9 2 1
5 8 5 3 8 4
n_H x n_W = 6 x 6
*
Filter
1 0 -1
1 0 -1
1 0 -1
=
Result
2 6
Parameters:
Size: f = 3
Stride: s = 1
Padding: p = 0
6 = 9*1 + 2*0 + 5*(-1) + 6*1 + 2*0 + 4*(-1) + 4*1 + 5*0 + 4*(-1)
https://indoml.com



$$\begin{bmatrix} 0&0&0&0&0&0&0&0\\ 0&4&9&2&5&8&3&0\\ 0&5&6&2&4&0&3&0\\ 0&2&4&5&4&5&2&0\\ 0&5&6&5&4&7&8&0\\ 0&5&7&7&9&2&1&0\\ 0&5&8&5&3&8&4&0\\ 0&0&0&0&0&0&0&0 \end{bmatrix} * \begin{bmatrix} 1&0&-1\\ 1&0&-1\\ 1&0&-1 \end{bmatrix} = \begin{bmatrix} -15&&\\ &&\\ &&\end{bmatrix}$$

$$\begin{aligned} &= 0*1 + 0*0 + 0*(-1) + \\ &\quad 0*1 + 4*0 + 9*(-1) + \\ &\quad 0*1 + 9*0 + 6*(-1) \\ &= -15 \end{aligned}$$

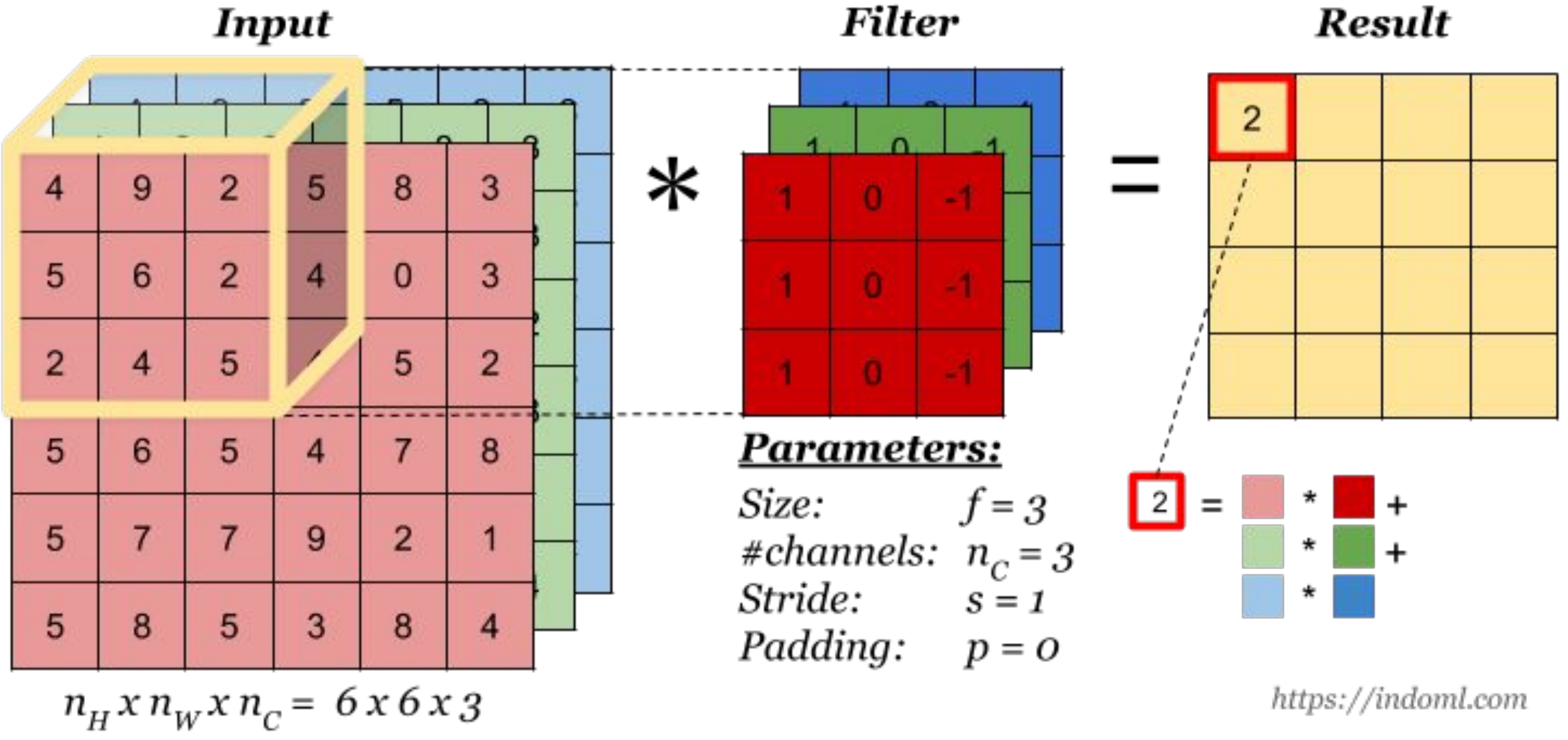
Output Dimension



Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

http://cs231n.github.io/convolutional-networks/

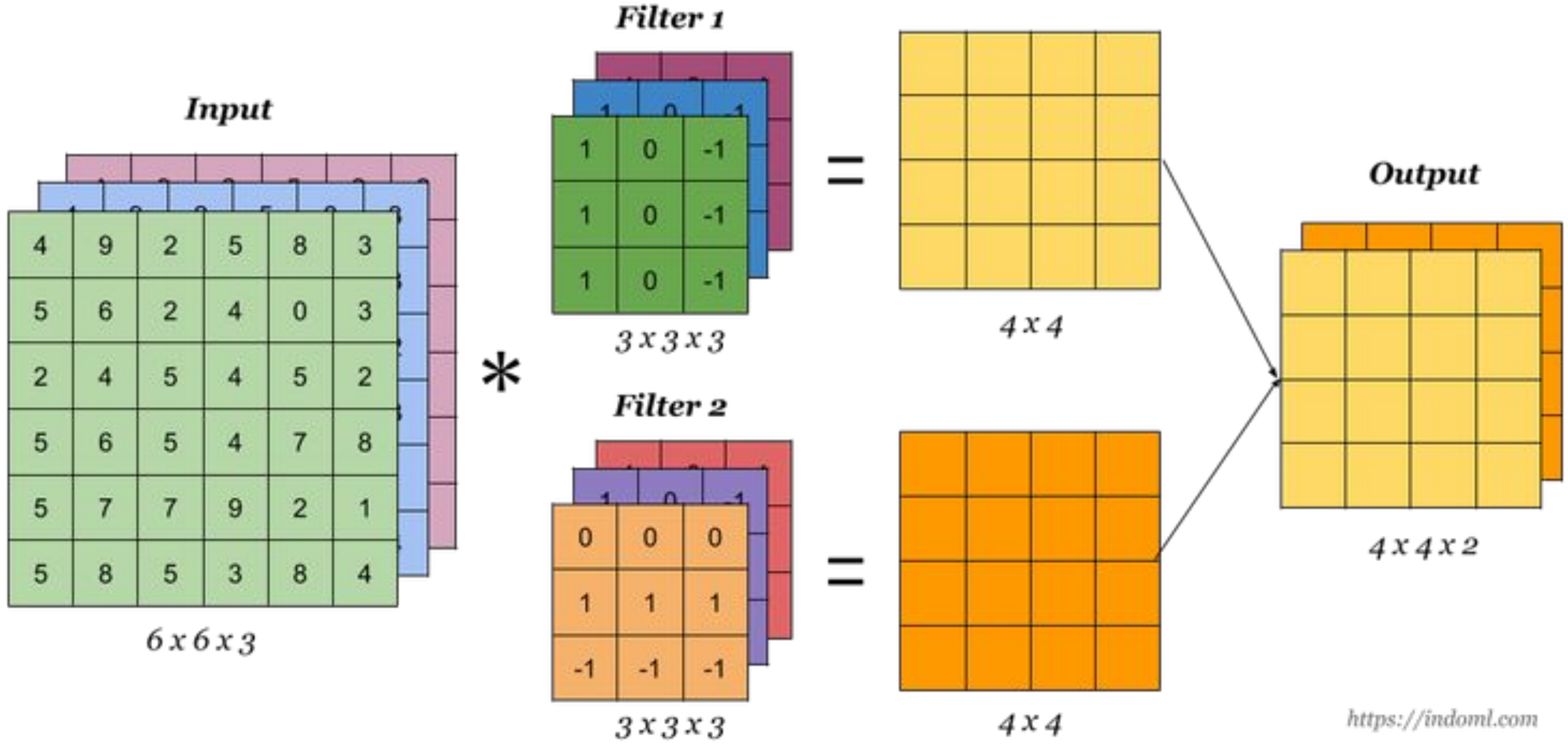


Parameters:

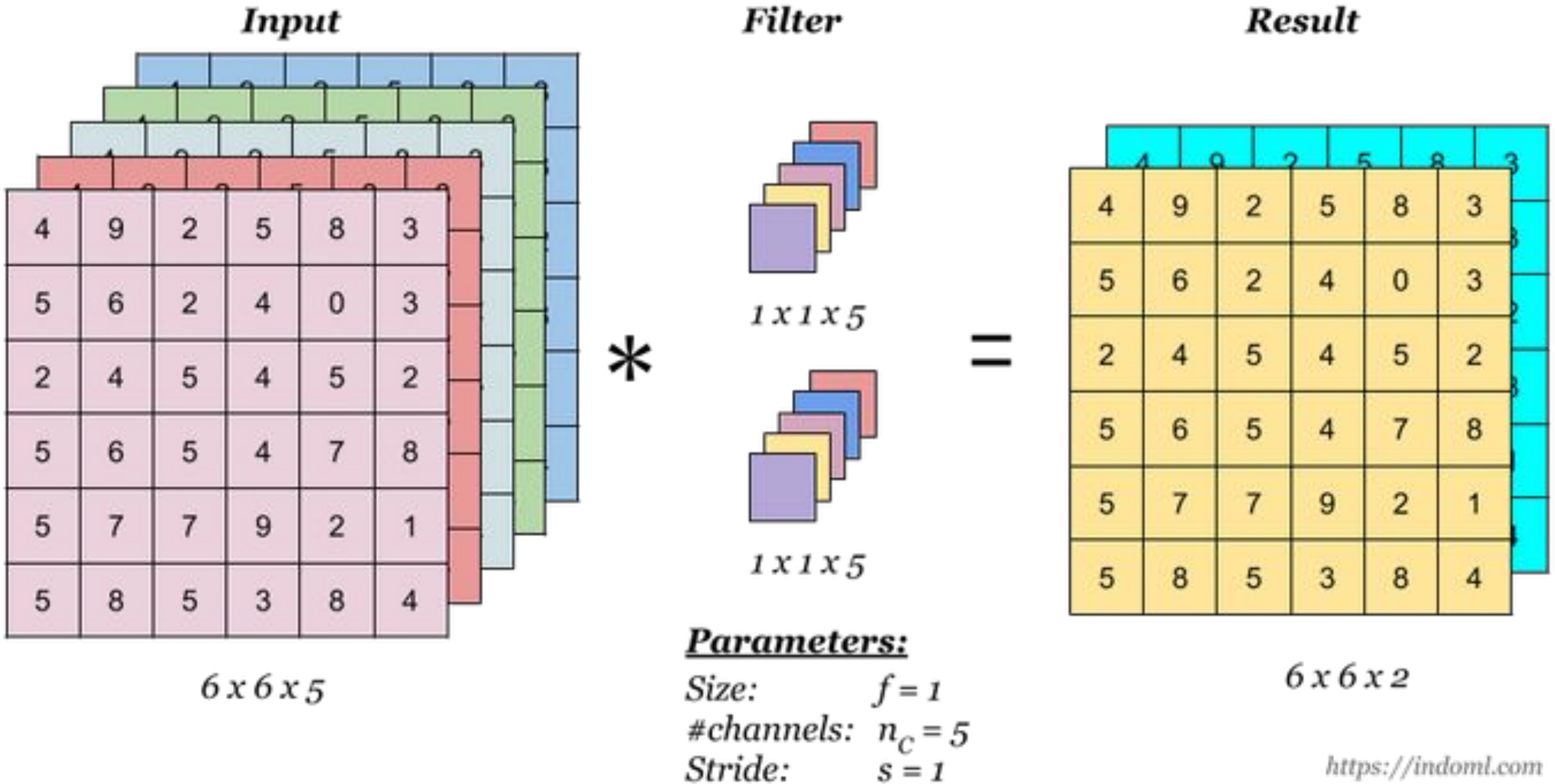
Size:	$f = 3$
#channels:	$n_C = 3$
Stride:	$s = 1$
Padding:	$p = 0$

https://indoml.com

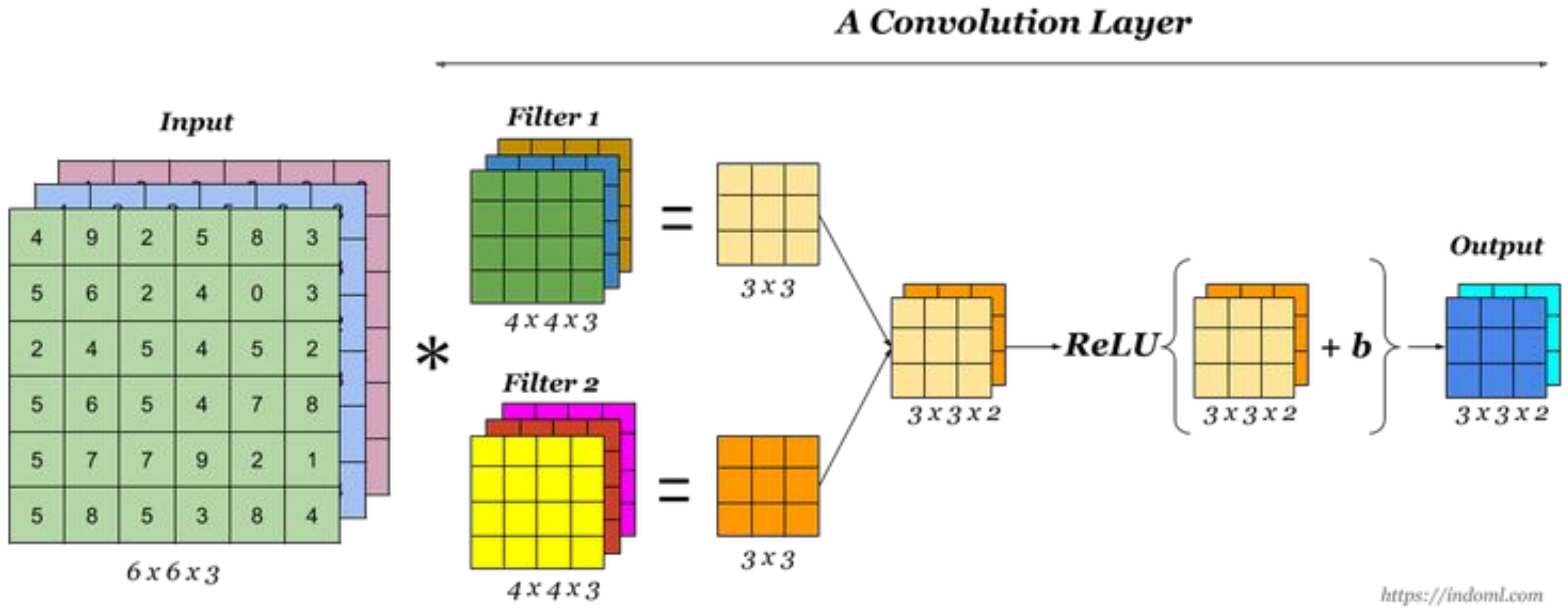
Convolution Operation with Multiple Filters



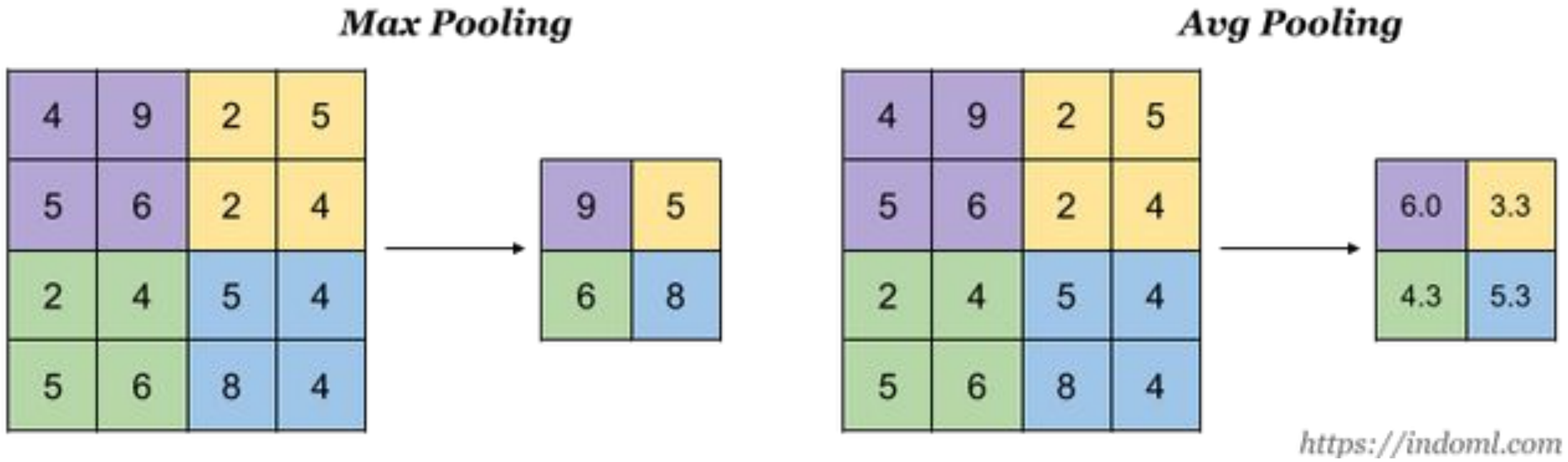
1 x 1 Convolution



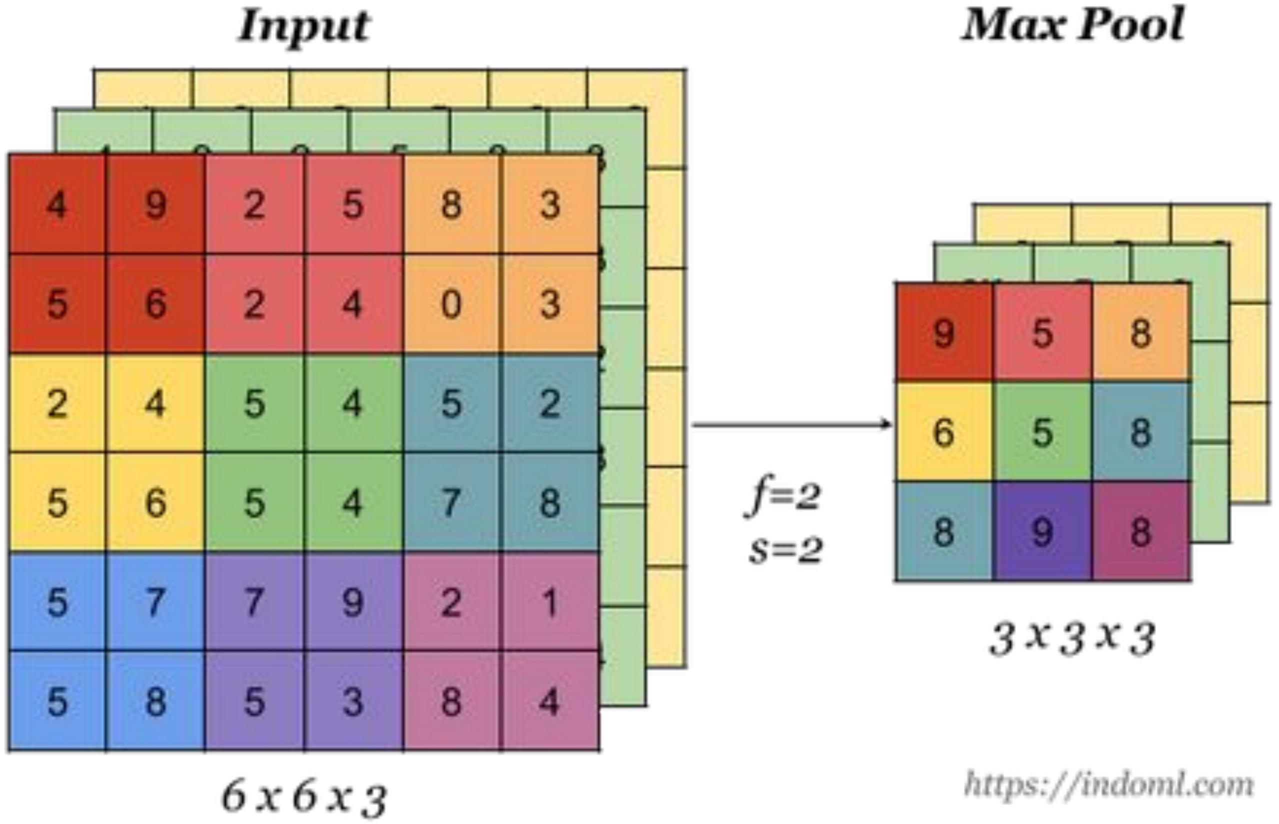
One Convolution Layer



Pooling Layer



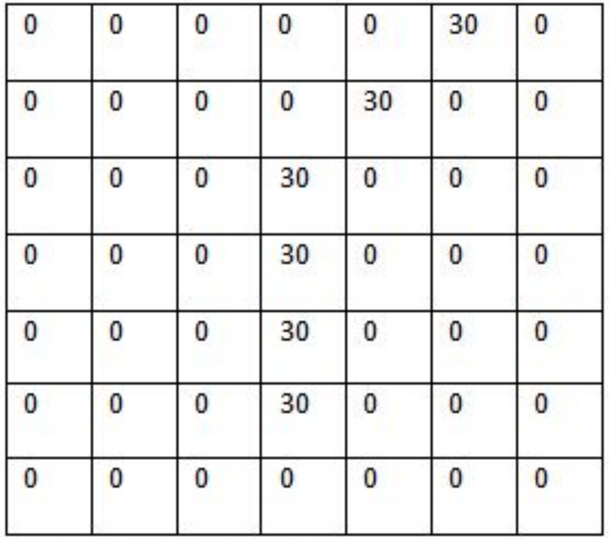
Max Pooling



Input
Max Pool
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
f=2
s=2
9 5 8
6 5 8
8 9 8
6 x 6 x 3
3 x 3 x 3
https://indoml.com

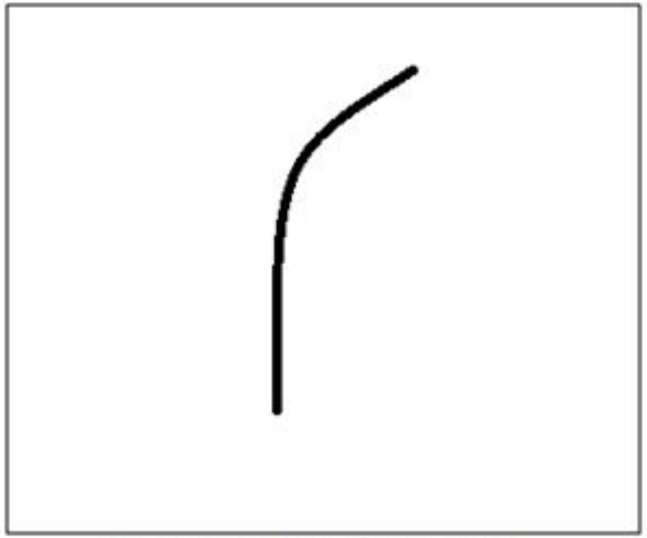
What this convolution is actually doing?

Features



0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

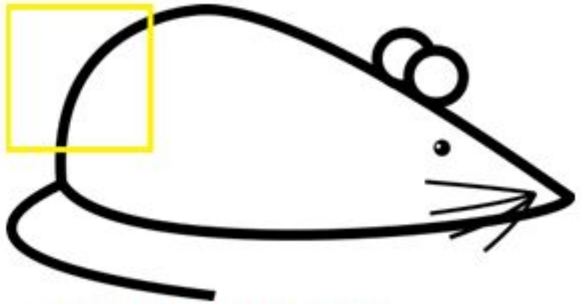
Pixel representation of filter

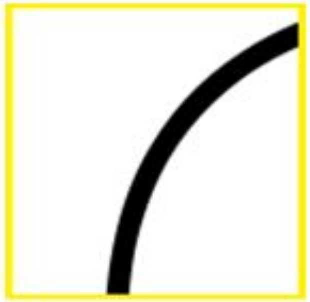


Visualization of a curve detector filter

https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

Original image


Visualization of the filter on the image


Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

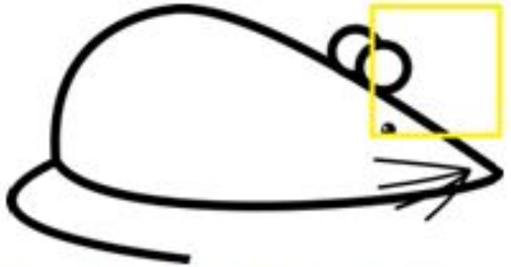
*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/



Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

*

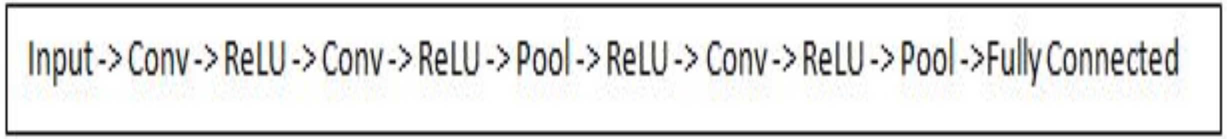
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = 0

https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

A classic CNN architecture would look like this.



Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool ->Fully Connected

What the different layers do?

- We talked about what the filters in the first conv layer are designed to detect. They detect low level features such as edges and curves.

- In order to predict whether an image is a type of object, we need the network to be able to recognize higher level features such as hands or paws or ears.

- When we go through another conv layer, the output of the first conv layer becomes the input of the 2nd conv layer.

- When we were talking about the first layer, the input was just the original image. However, when we're talking about the 2nd conv layer, the input is the activation map(s) that result from the first layer.

- So each layer of the input is basically describing the locations in the original image for where certain low level features appear.

- Now when you apply a set of filters on top of that (pass it through the 2nd conv layer), the output will be activations that represent higher level features.

- Types of these features could be semicircles (combination of a curve and straight edge) or squares (combination of several straight edges)

- As you go through the network and go through more conv layers, you get activation maps that represent more and more complex features

Fully connected layer basically takes an input volume (whatever the output is of the conv or ReLU or pool layer preceding it) and outputs an N dimensional vector where N is the number of classes http://scs.ryerson.ca/~aharley/vis/conv/flat.html

The image

A	B	C
D	E	F
G	H	J

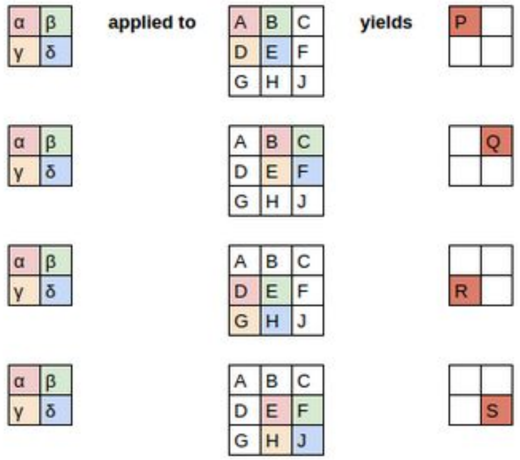
The filter

α	β
γ	δ

Since the filter fits in the image four times, we have four results

P	Q
R	S

Here's how we applied the filter to each section of the image to yield each result



The equation view

$$\boxed{\alpha} * \boxed{A} + \boxed{\beta} * \boxed{B} + \boxed{\gamma} * \boxed{D} + \boxed{\delta} * \boxed{E} + b = \boxed{P}$$

$$\boxed{\alpha} * \boxed{B} + \boxed{\beta} * \boxed{C} + \boxed{\gamma} * \boxed{E} + \boxed{\delta} * \boxed{F} + b = \boxed{Q}$$

$$\boxed{\alpha} * \boxed{D} + \boxed{\beta} * \boxed{E} + \boxed{\gamma} * \boxed{G} + \boxed{\delta} * \boxed{H} + b = \boxed{R}$$

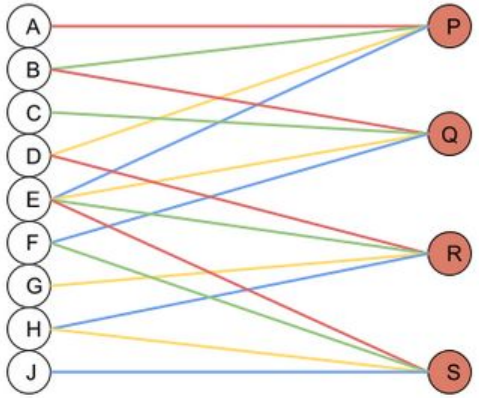
$$\boxed{\alpha} * \boxed{E} + \boxed{\beta} * \boxed{F} + \boxed{\gamma} * \boxed{H} + \boxed{\delta} * \boxed{J} + b = \boxed{S}$$

Notice that the bias term, b, is the same for each section of the image. You can consider the bias as part of the filter, just like the weights (α, β, γ, δ) are part of the filter.

The compact equation view

$$\begin{aligned} \alpha A+\beta B+\gamma D+\delta E+b &= P \\ \alpha B+\beta C+\gamma E+\delta F+b &= Q \\ \alpha D+\beta E+\gamma G+\delta H+b &= R \\ \alpha E+\beta F+\gamma H+\delta J+b &= S \end{aligned}$$

The neural network view



The matrix multiplication view

$$\begin{array}{c}\begin{bmatrix} \alpha & \beta & 0 & \gamma & \delta & 0 & 0 & 0 & 0 \\ 0 & \alpha & \beta & 0 & \gamma & \delta & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha & \beta & 0 & \gamma & \delta & 0 \\ 0 & 0 & 0 & 0 & \alpha & \beta & 0 & \gamma & \delta \end{bmatrix} \\ \begin{array}{ccccccccc} A & B & C & D & E & F & G & H & J \end{array}\end{array} * \begin{bmatrix} A \\ B \\ C \\ D \\ E \\ F \\ G \\ H \\ J \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \\ b \end{bmatrix} = \begin{bmatrix} \alpha A+\beta B+0C+\gamma D+\delta E+0F+0G+0H+0J+b \\ 0A+\alpha B+\beta C+0D+\gamma E+\delta F+0G+0H+0J+b \\ 0A+0B+0C+\alpha D+\beta E+0F+\gamma G+\delta H+0J+b \\ 0A+0B+0C+0D+\alpha E+\beta F+0G+\gamma H+\delta J+b \end{bmatrix} = \begin{bmatrix} \alpha A+\beta B+\gamma D+\delta E+b \\ \alpha B+\beta C+\gamma E+\delta F+b \\ \alpha D+\beta E+\gamma G+\delta H+b \\ \alpha E+\beta F+\gamma H+\delta J+b \end{bmatrix} = \begin{bmatrix} P \\ Q \\ R \\ S \end{bmatrix}$$

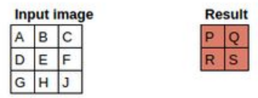
The matrix above is a weight matrix, just like the ones from traditional neural networks. However, this weight matrix has two special properties:

1. The zeros shown in gray are untrainable. This means that they'll stay zero throughout the optimization process.
2. Some of the weights are equal, and while they are trainable (i.e. changeable), they must remain equal. These are called "shared weights".

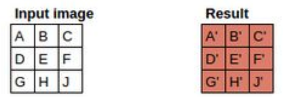
The zeros correspond to the pixels that the filter didn't touch. Each row of the weight matrix corresponds to one application of the filter.

Bonus: Preserving the image size with zero-padding and a 3x3 filter

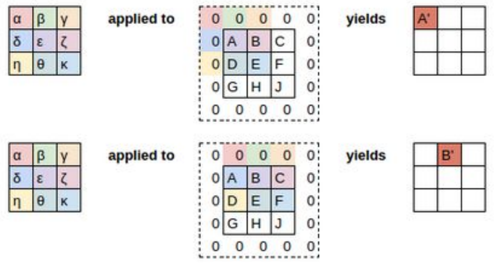
Notice that we went from a 3x3 image to a 2x2 image.



With zero-padding and a 3x3 filter, we can preserve the image size.

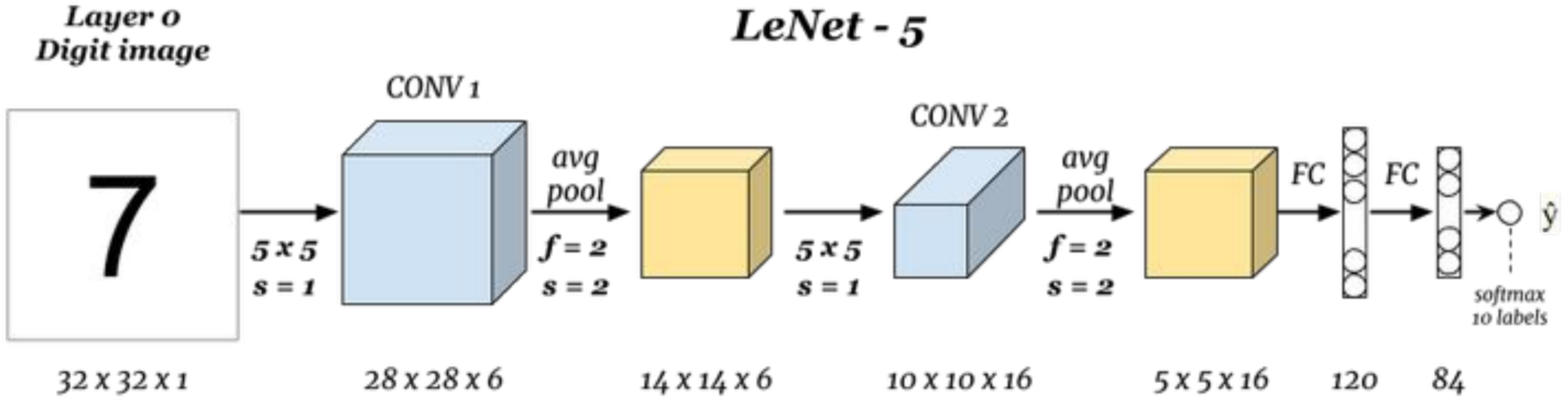


Here's what zero-padding with a 3x3 filter looks like:

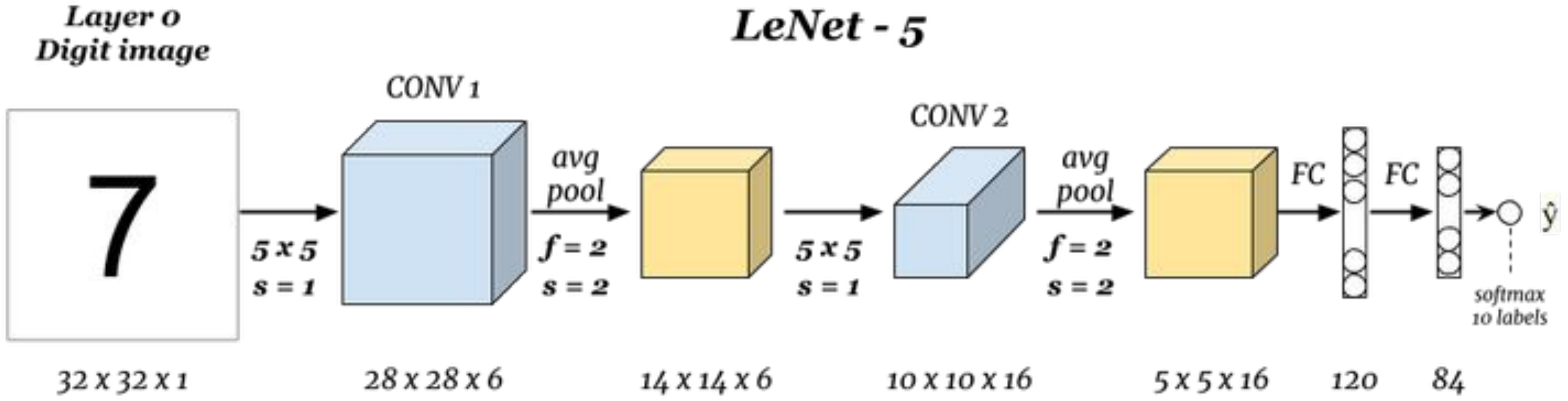


The 3x3 filter allows us to center the filter on each pixel, so that each original pixel corresponds to a pixel in the result. Although, without zero-padding, our filter would get stuck in the corners, unable to center on any pixel except pixel E, and the result would be a 1x1 image.

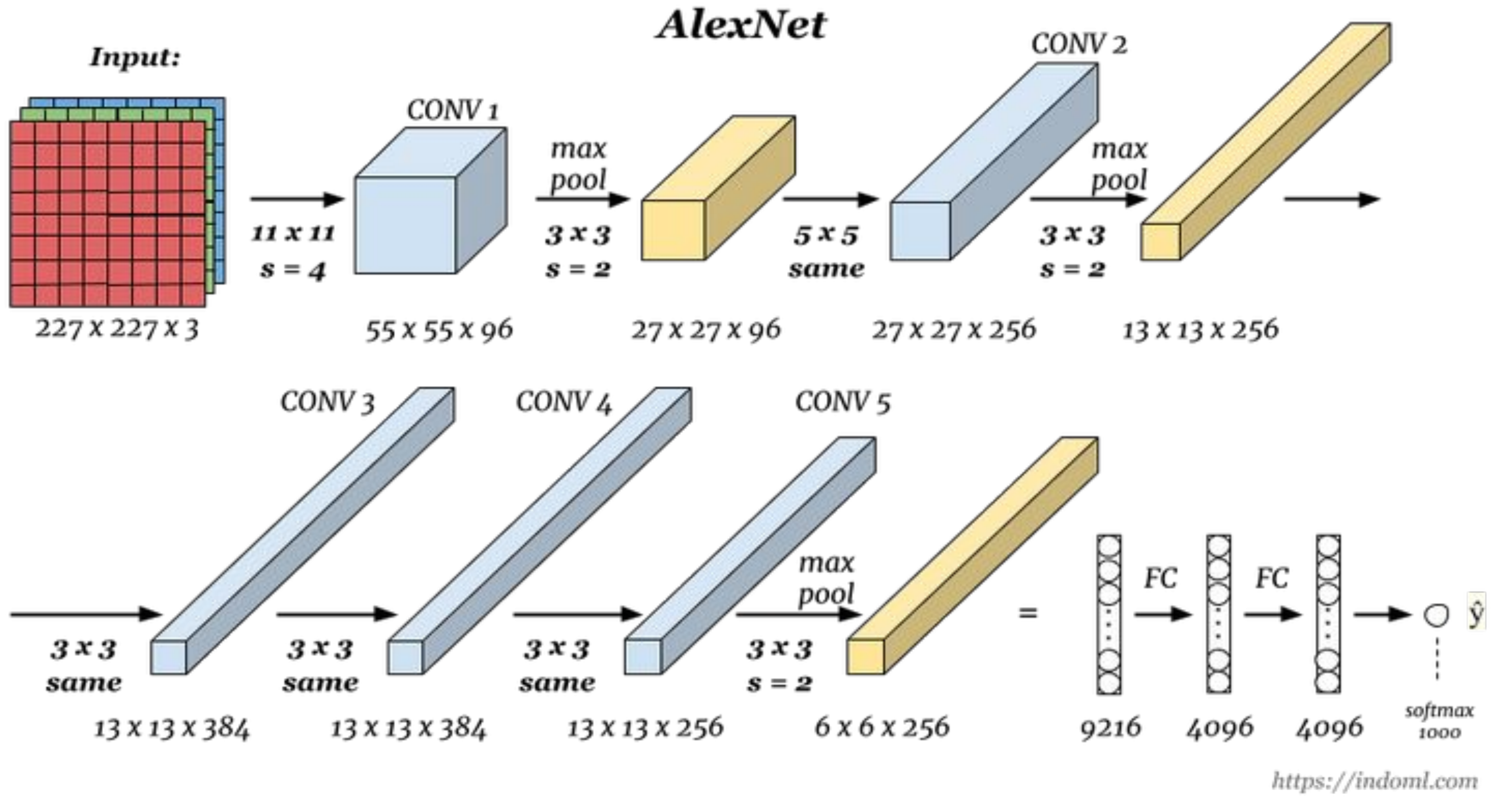
Classic Network: LeNet – 5



Classic Network: LeNet – 5



Classic Network: AlexNet



HANDS ON:

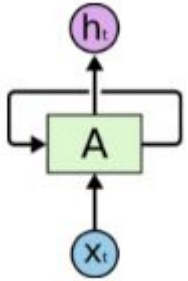
CNN on MNIST

https://colab.research.google.com/drive/1JI7LkJd2LESDm-nnWFT2yyBXgYnhI2oF

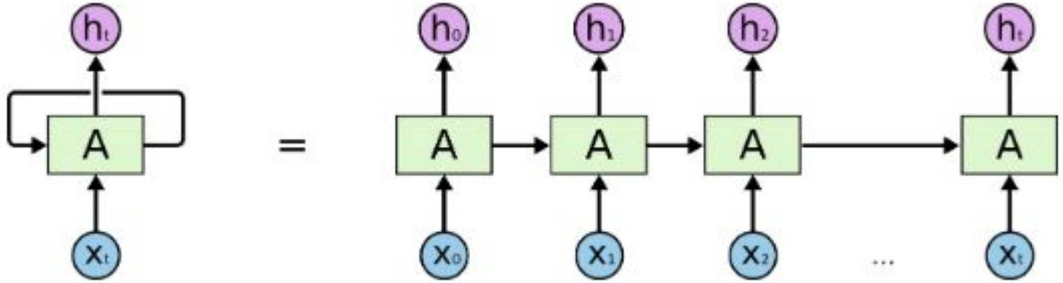
https://bit.ly/2AT2RUG

Recurrent Neural Networks (RNN)

- In a traditional neural network we assume that all inputs (and outputs) are independent of each other
- For some tasks that’s a very bad idea
- If you want to predict the next word in a sentence you better know which words came before it
- RNNs are called *recurrent* because they perform the same task for every element of a sequence, with the output being depended on the previous computations
- Another way to think about RNNs is that they have a “memory” which captures information about what has been calculated so far.

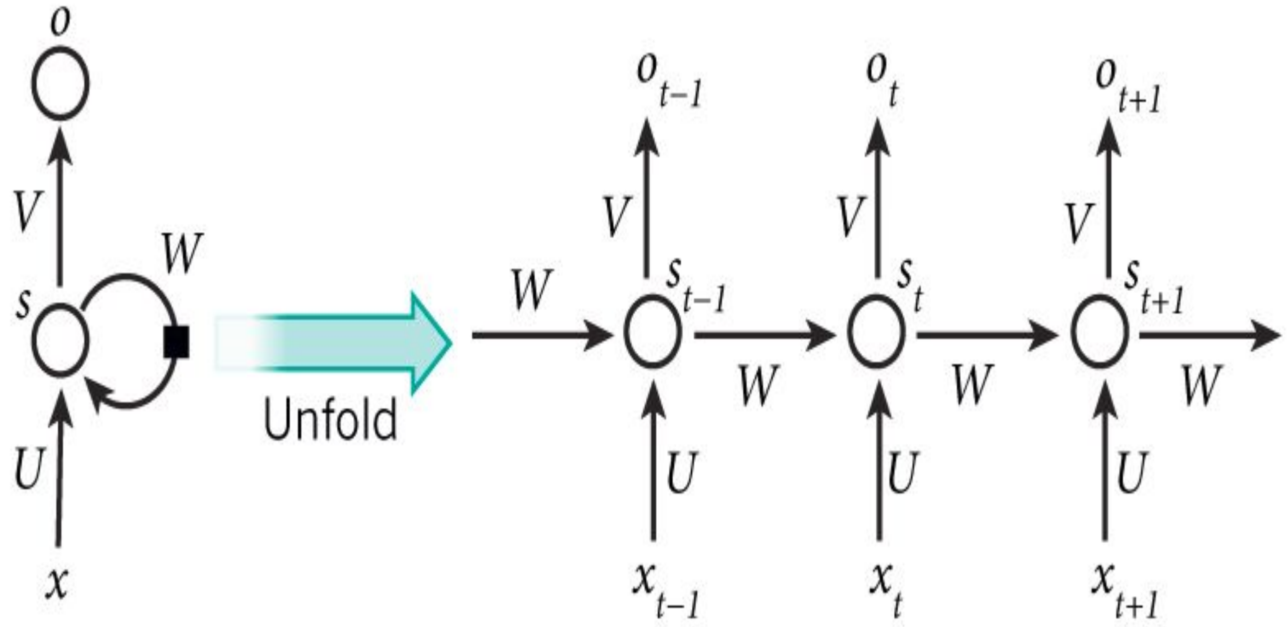


Recurrent Neural Networks have loops.

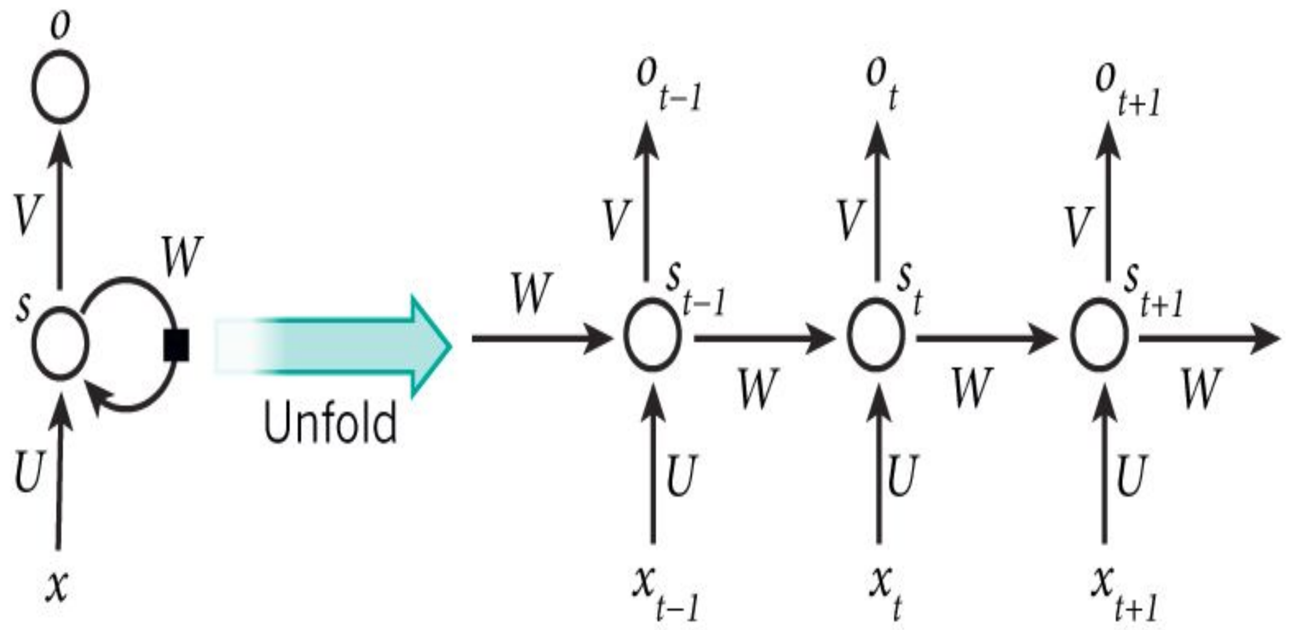


An unrolled recurrent neural network.

http://colah.github.io/posts/2015-08-Understanding-LSTMs/



A recurrent neural network and the unfolding in time of the computation involved in its forward computation. Source: Nature



A recurrent neural network and the unfolding in time of the computation involved in its forward computation. Source: Nature

$$S_t = f(U\,x_t + W\,s_{t-1})$$
$$O_t = g(V\,x_t\,)$$

What can RNNs do?

Machine Translation

Input is a sequence of words in our source language (e.g. German), We want to output a sequence of words in our target language (e.g. English).

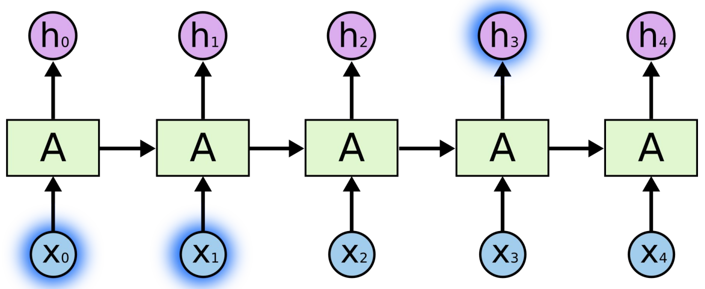
Speech Recognition

Given an input sequence of acoustic signals from a sound wave, we can predict a sequence of phonetic segments together with their probabilities

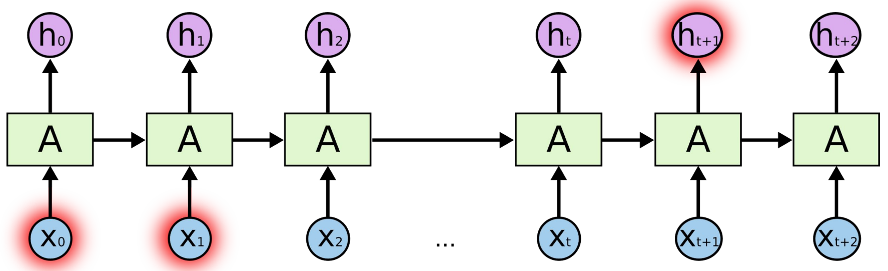
Long Short Term Memory networks (LSTM)

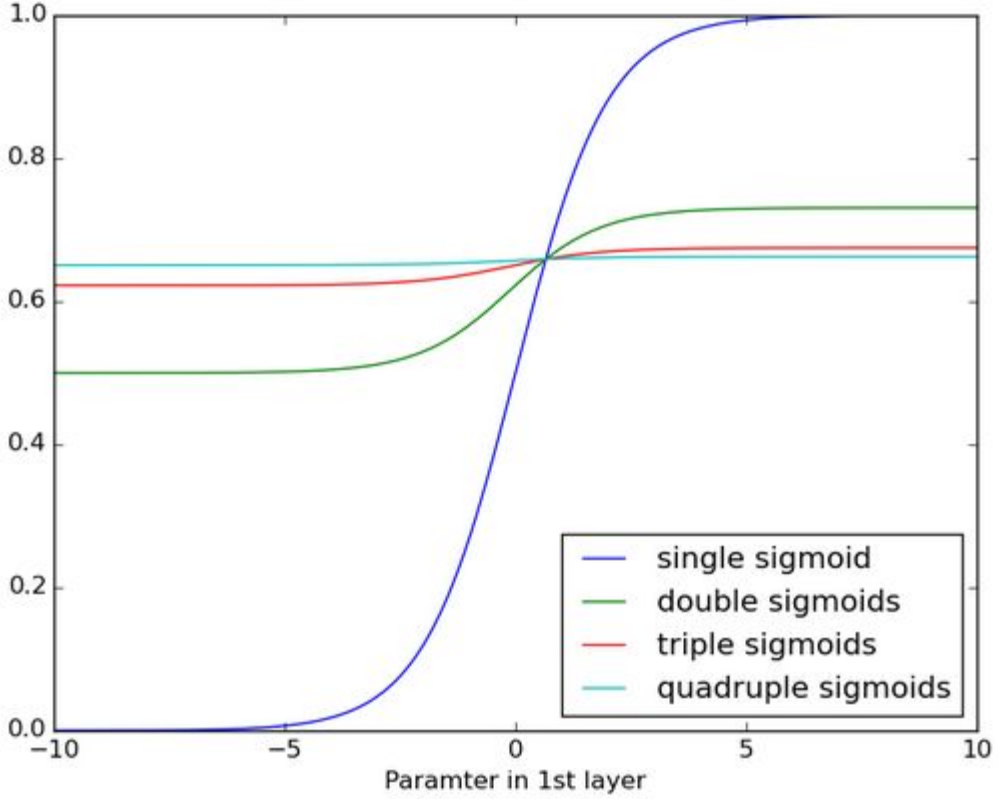
RNNs -The Problem of Long-Term Dependencies and Vanishing Gradients

“the clouds are in the *sky*,”



“I grew up in France…….. I speak fluent *French*.”

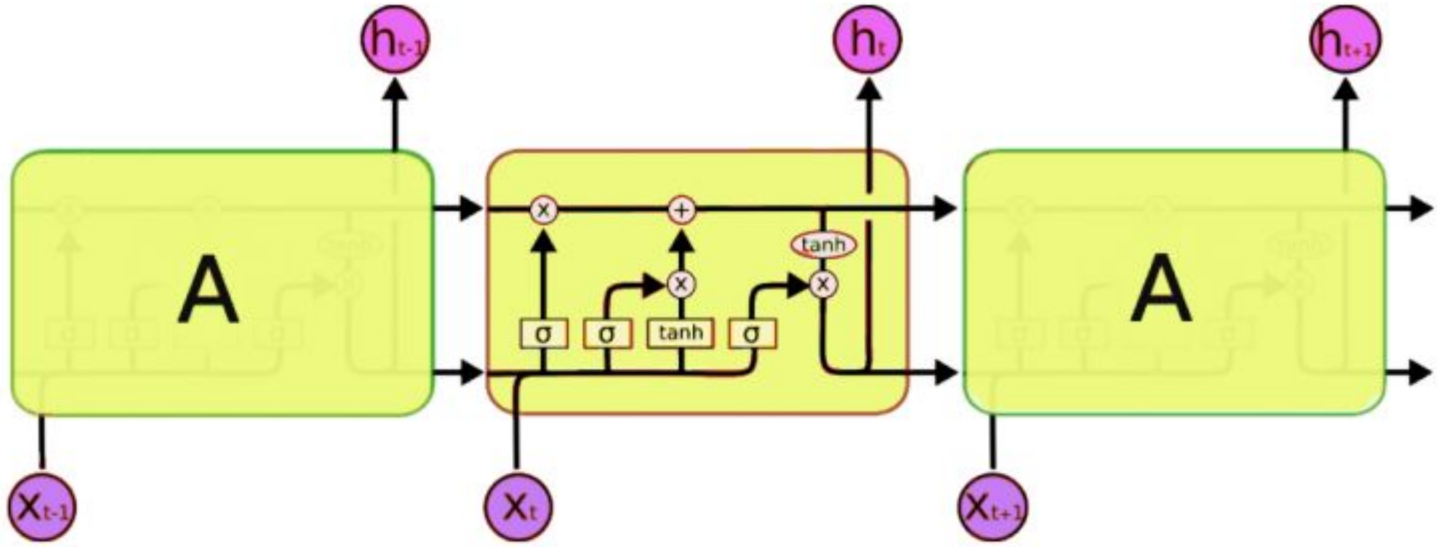




https://skymind.ai/wiki/lstm

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

Architecture of LSTMs



The functioning of LSTM can be visualized by understanding the functioning of a news channel's team covering a murder story. Now, a news story is built around facts, evidence and statements of many people. Whenever a new event occurs you take either of the three steps.

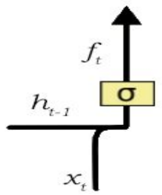
Let's say, we were assuming that the murder was done by 'poisoning' the victim, but the autopsy report that just came in said that the cause of death was 'an impact on the head'. Being a part of this news team what do you do? You immediately **forget** the previous cause of death and all stories that were woven around this fact.

What, if an entirely new suspect is introduced into the picture. A person who had grudges with the victim and could be the murderer? You **input** this information into your news feed, right?

Now all these broken pieces of information cannot be served on mainstream media. So, after a certain time interval, you need to summarize this information and **output** the relevant things to your audience. Maybe in the form of "*XYZ turns out to be the prime suspect.*".

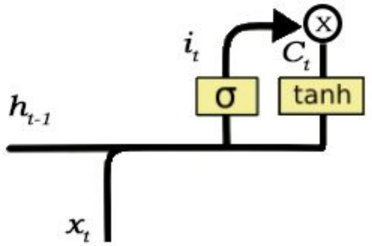
Forget Gate

Bob is a nice person. Dan on the other hand is evil.



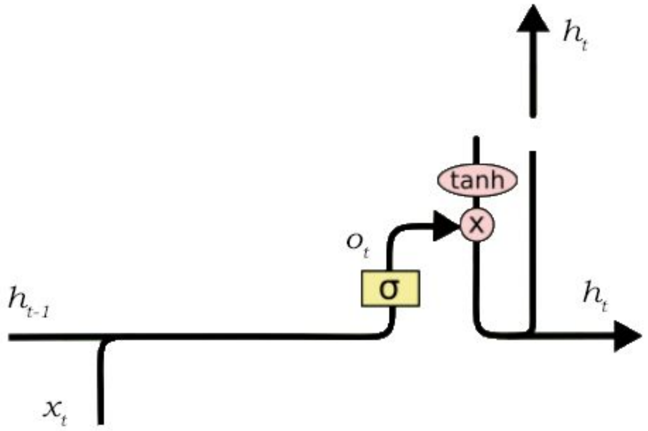
Input Gate

Bob knows swimming. He told me over the phone that he had served the navy for 4 long years.

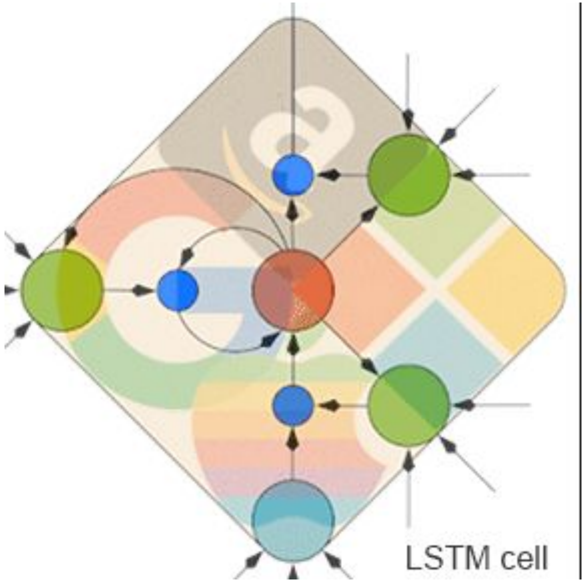


Output Gate

Bob fought single handedly with the enemy and died for his country. For his contributions brave ______.



- Speech recognition on over 2 billion Android phones (since mid 2015)
- Greatly improved machine translation through Google Translate (since Nov 2016)
- Facebook (over 4 billion LSTM-based translations per day as of 2017)
- Siri and Quicktype on almost 1 billion iPhones (since 2016)
- The answers of Amazon's Alexa, and numerous other applications



Acknowledgment

- Most of the contents of this presentation are taken from different sources.
- This presentation is used for teaching purpose only.
- It is difficult to cite different sources. We are thankful to all the people whose work/slides helped to contribute to this presentation.