université de BORDEAUX

Convolutional Neural Network

LE Van Linh

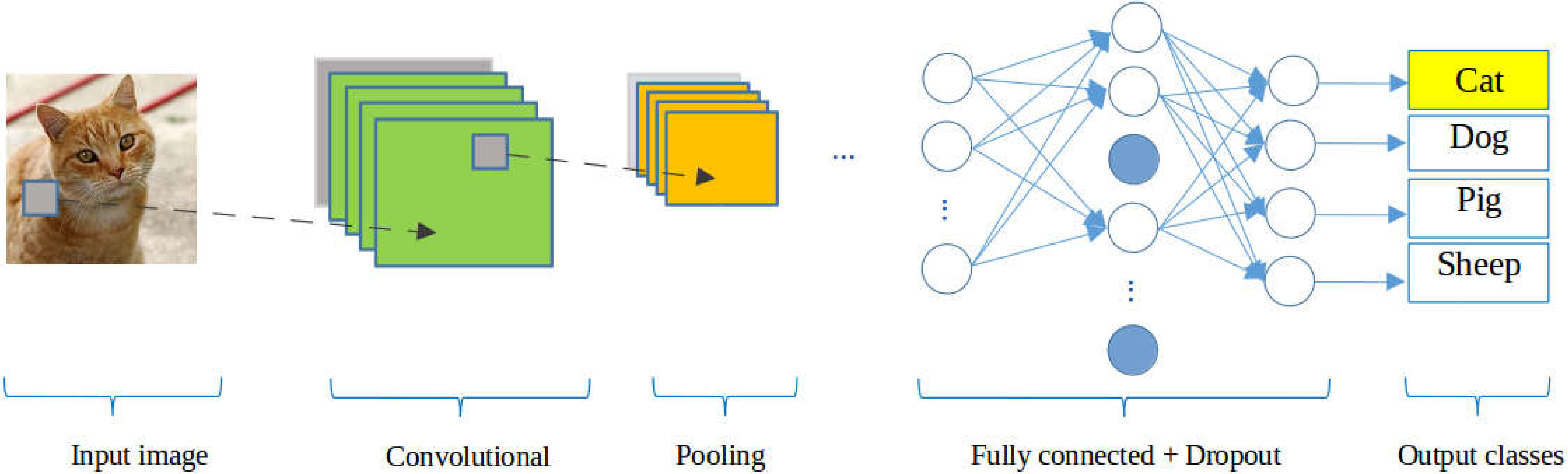
van-linh.le@u-Bordeaux.fr

http://www.labri.fr/perso/vle

Contents

1. Architecture overview
2. Convolutional Neural Network’s Layers
3. Case Studies
4. Implement a CNN in PyTorch

Architecture overview



Architecture overview

Similar with ordinary Neural Network:

- Make up of neurons that have learnable weights and bias
- Each neuron receives some inputs, perform a dot product and optionally follows it with a non-linearity function.
- Use different kinds of activation functions

However:

- The input are images
- Encode certain properties into the architecture
- Reduce the amount of parameters in the network
- Arrange in 3 dimensions: width, height, depth

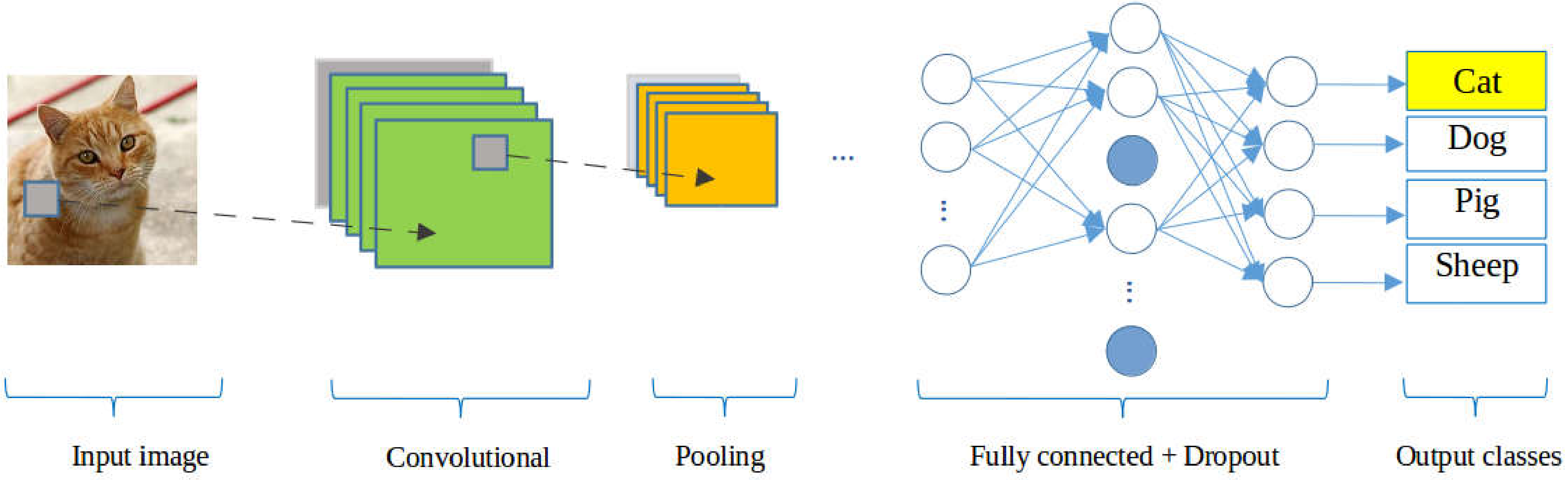
Contents

1. Architecture overview
2. **Convolutional Neural Network’s Layers**
3. Case Studies
4. Implement a CNN in PyTorch

Convolutional Neural Network’s Layers

Three main types of layers to build CNN architectures:

- Convolutional Layer
- Pooling layer
- Fully-Connected layer



Convolutional Neural Network's Layers

Convolutional Layer: is the core of CNN with the parameters consist of a set of learnable filters

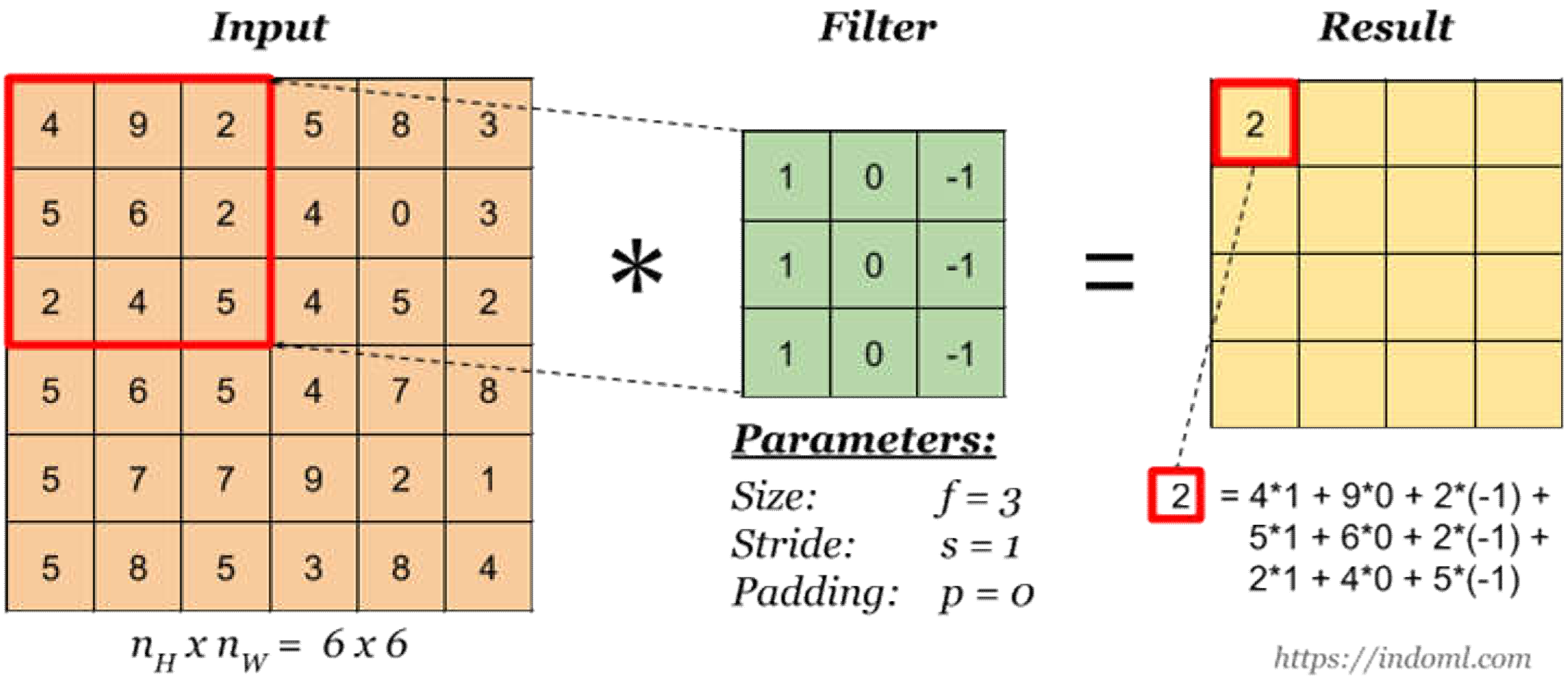
- Accept a input of $(W_1 \times H_1 \times D_1)$
- Hyper-parameters: Number of filters K, filter size F, the stride S, amount of zero padding P
- Output with the size of $(W_2 \times H_2 \times D_2)$, where:

$$W_2 = (W_1 - F + 2P)/S + 1$$

$$H_2 = (H_1 - F + 2P)/S + 1$$

$$D_2 = K$$

Convolutional Neural Network's Layers



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

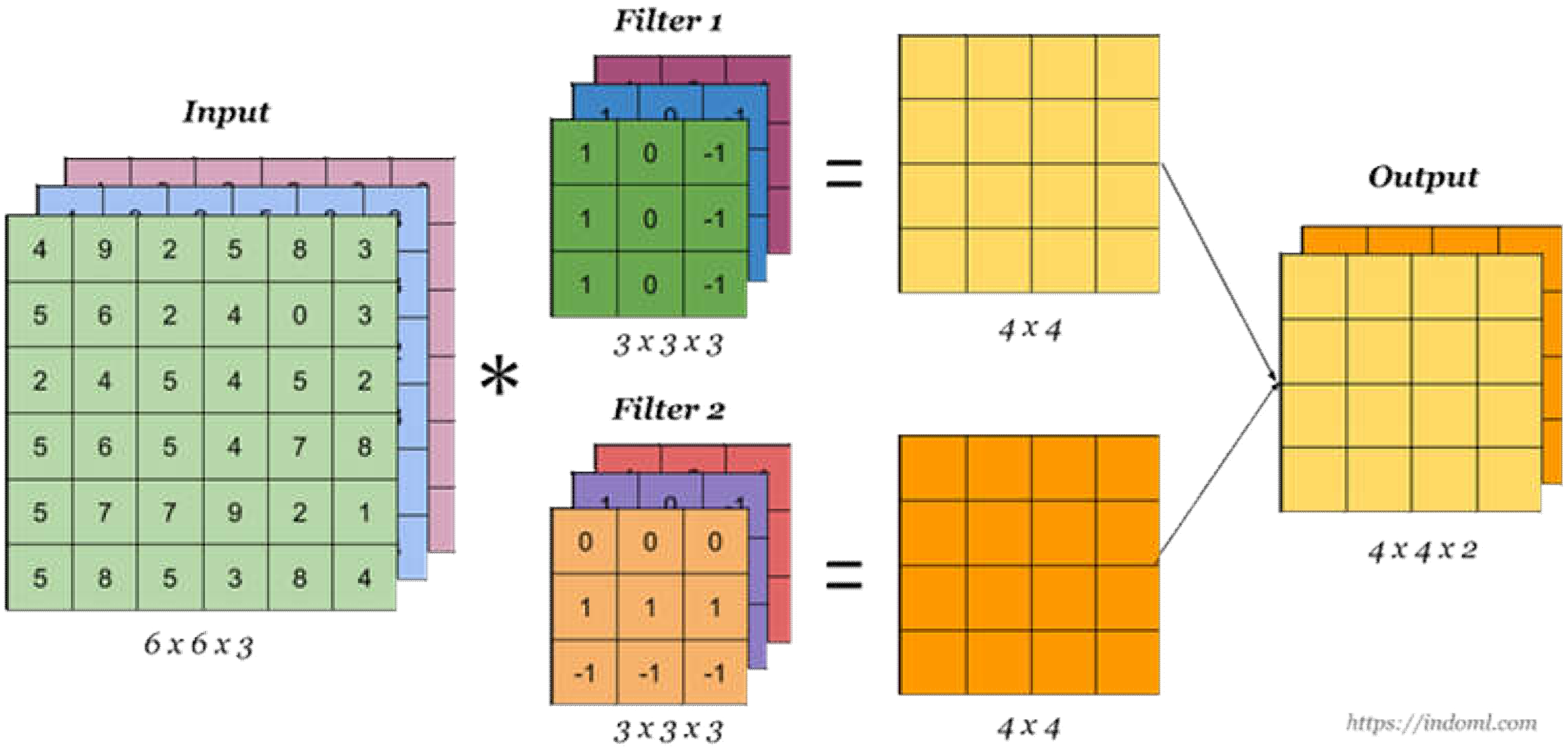
2			

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

https://indoml.com

Image from indoml.com

Convolutional Neural Network’s Layers



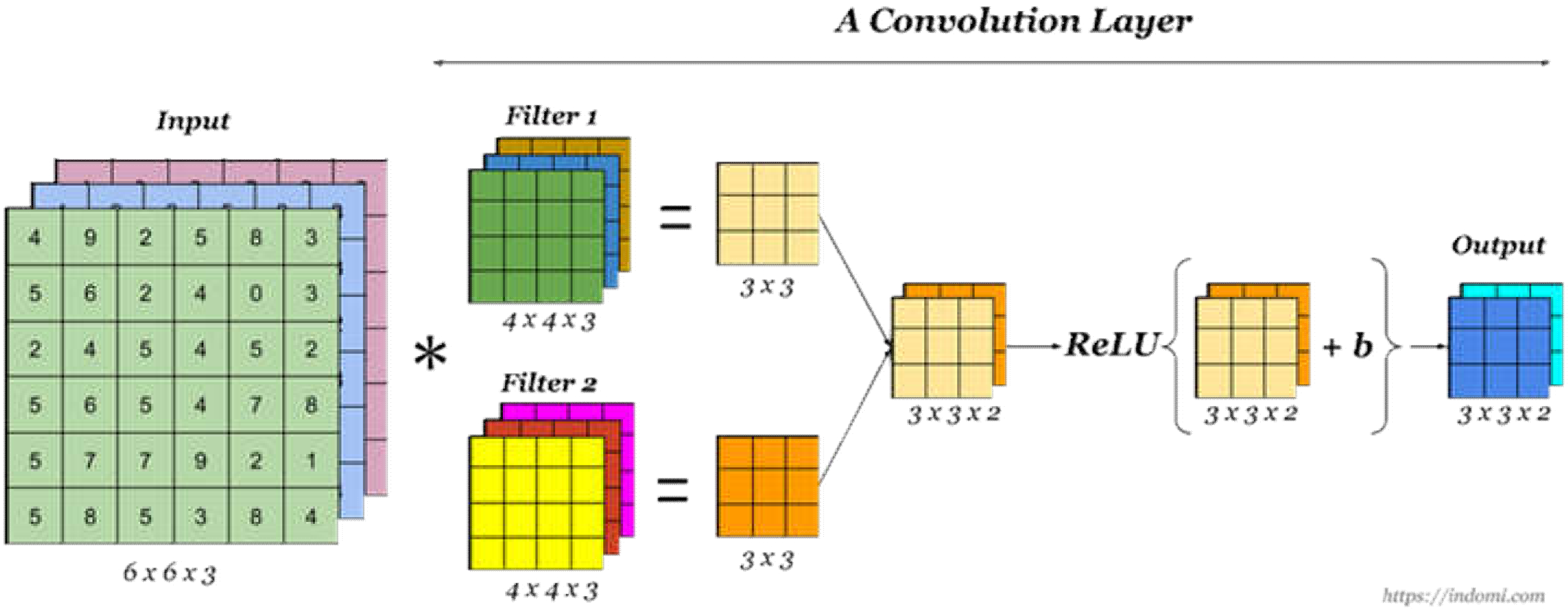
4	9	2	5	8	3
5	6	2	4	0	3
2	4	5	4	5	2
5	6	5	4	7	8
5	7	7	9	2	1
5	8	5	3	8	4

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

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

Image from indoml.com

Convolutional Neural Network’s Layers



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

https://indomi.com

Image from indoml.com

Convolutional Neural Network's Layers

Pooling layer:

- Reduce the spatial size of the representation
- Reduce the amount of parameters and computation
- Control overfitting
- Two common types of pooling: MAX and AVERAGE

Convolutional Neural Network’s Layers

Pooling layer:

- Accept a input of $(W_1 \times H_1 \times D_1)$
- Hyper-parameters: filter size F, the stride S,
- Output with the size of $(W_2 \times H_2 \times D_2)$, where:

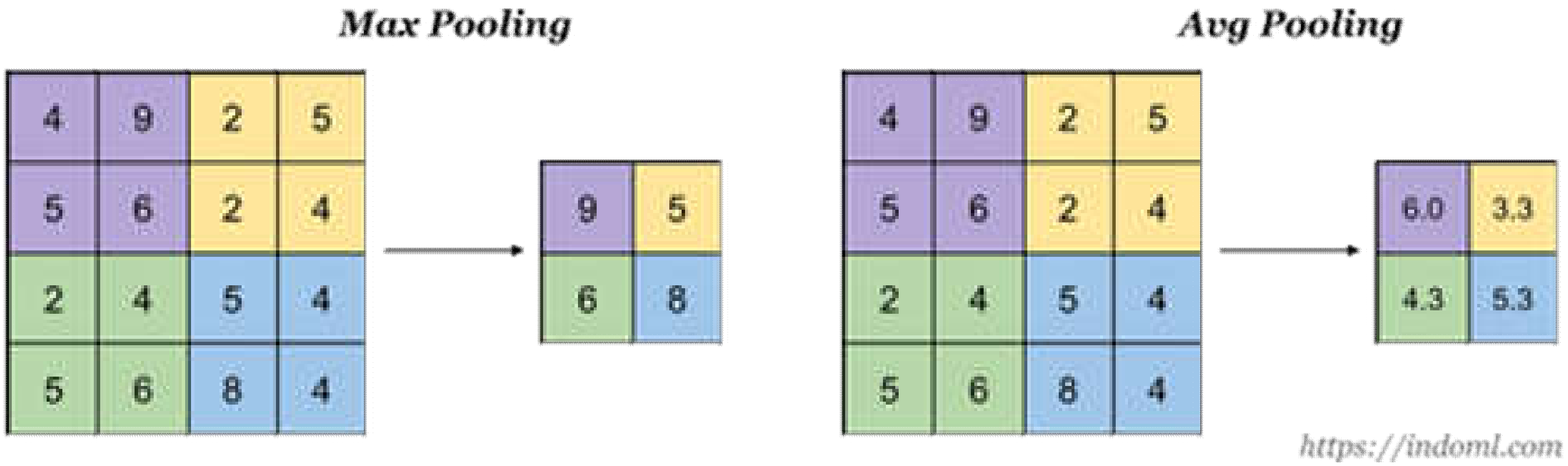
$$W_2 = (W_1 - F)/S + 1$$

$$H_2 = (H_1 - F)/S + 1$$

$$D_2 = D_1$$

The pooling layer have **zero** parameters

Convolutional Neural Network's Layers



Max Pooling

4	9	2	5
5	6	2	4
2	4	5	4
5	6	8	4

9	5
6	8

Avg Pooling

4	9	2	5
5	6	2	4
2	4	5	4
5	6	8	4

6.0	3.3
4.3	5.3

https://indoml.com

Image from indoml.com

Convolutional Neural Network’s Layers

Fully-connected layer:

- Have full connections to all activations in previous layer
- Their computation is the same as in regular Neural Network

Contents

1. Architecture overview
2. Convolutional Neural Network’s Layers
3. **Case Studies**
4. Implement a CNN in PyTorch

Case studies

LeNet-5: ~ 60 thousands parameters

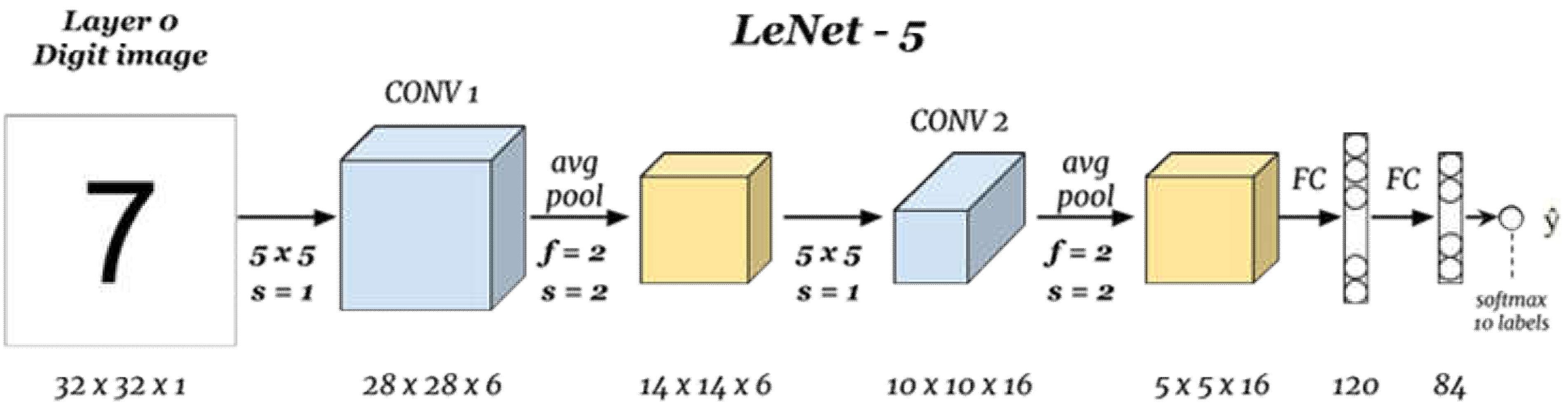


Image from indoml.com

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

Case studies

AlexNet: ~ 60 millions parameters

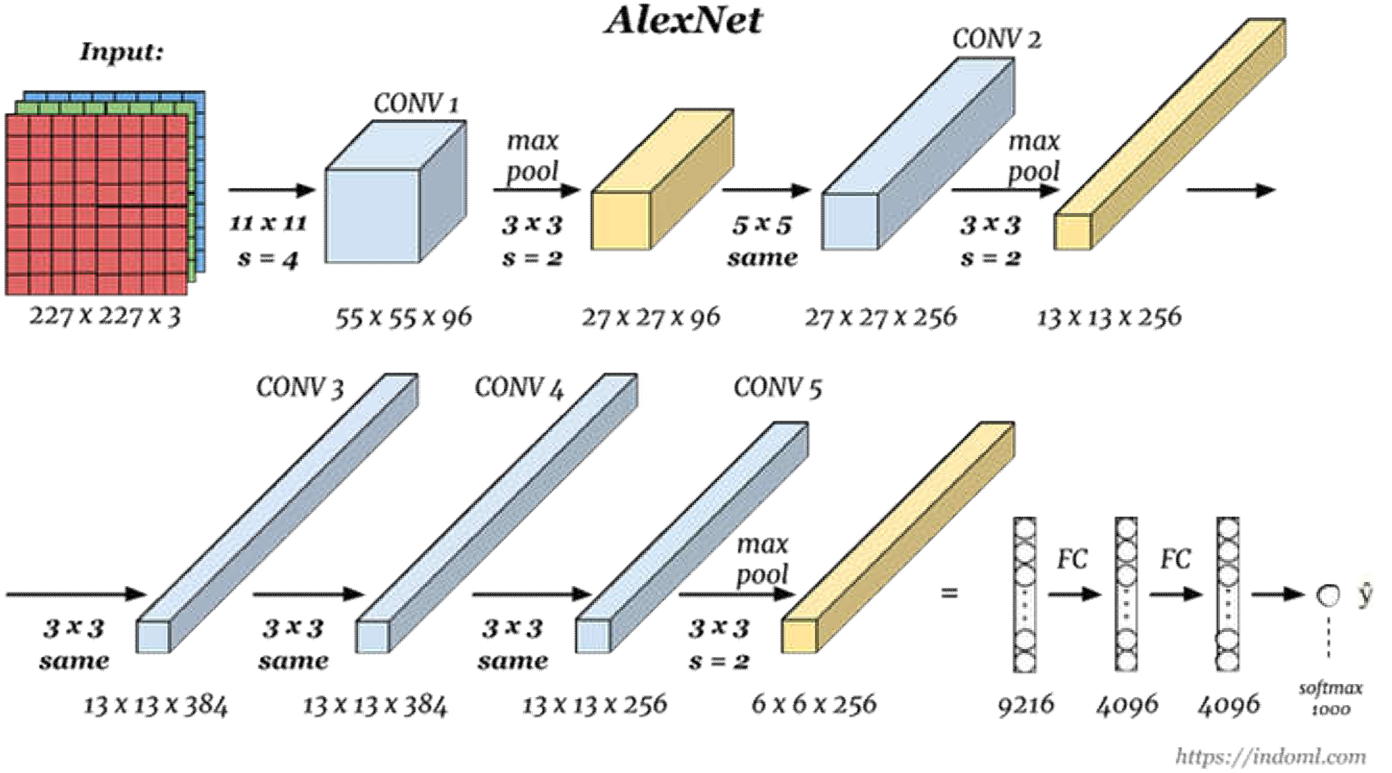


Image from indoml.com

Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever, *Classification with Deep Convolutional Neural Networks*, 2012

Case studies

VGG-16: ~ 38 millions parameters

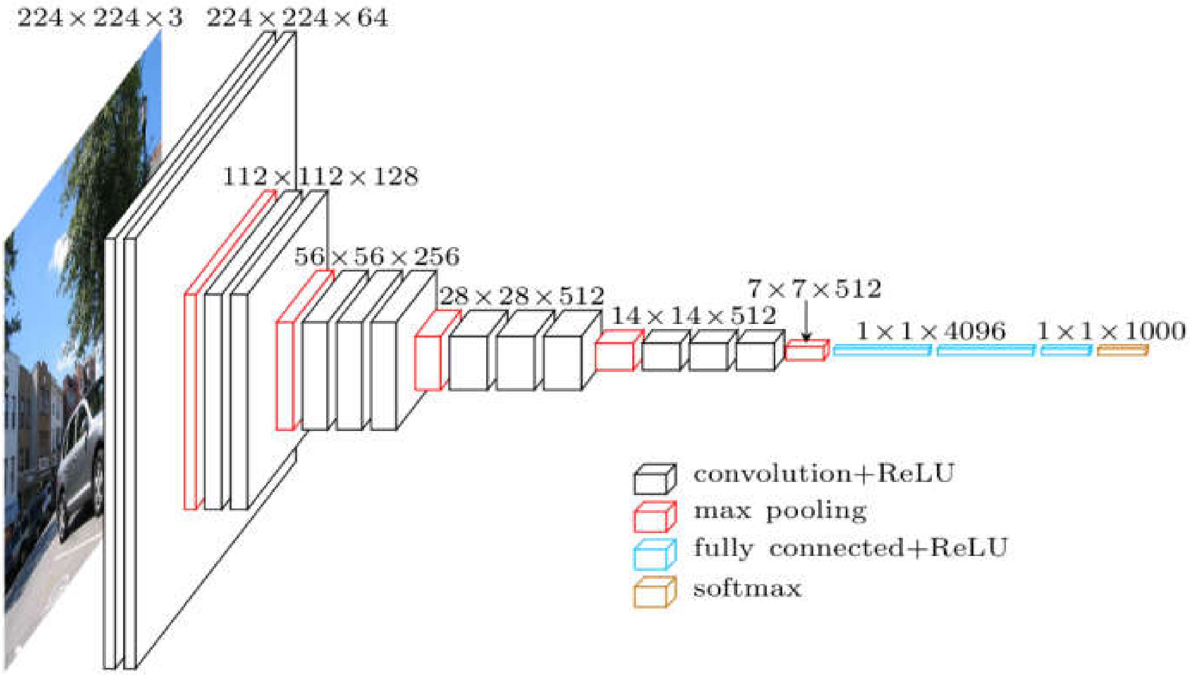


Image from blog.heuritech.com

Karen Simonyan and Andrew Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, 2014

Case studies

ResNet: use skip connections

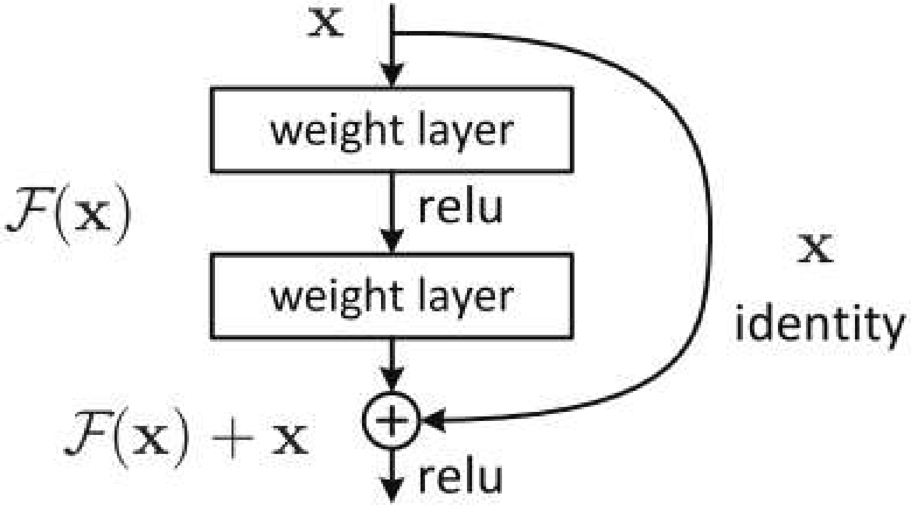
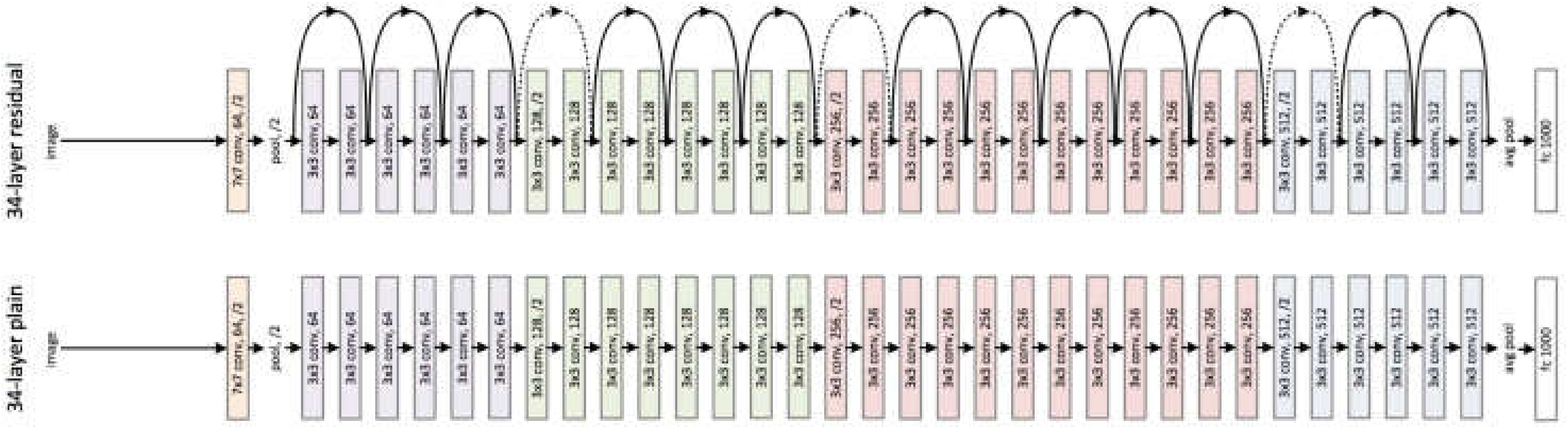

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Image from medium.com

He at al, *Deep Residual Learning for Image Recognition paper* , 2015/ Image from medium.com

Case studies

ResNet: use skip connections



He at al, *Deep Residual Learning for Image Recognition paper* , 2015/ Image from euler.stat.yale.edu

Contents

1. Architecture overview

2. Convolutional Neural Network’s Layers

3. Case Studies

4. **Implement a CNN in PyTorch**

Implement a CNN in PyTorch

Preparer:

- Data: MNIST for example
- Model: How many CONV layers, POOL layers, FC layers
- Hyper-parameters: number of epochs, loss functions, ….

Process:

- Like the neural network
- Use different kinds of layers in PyTorch library

université
de BORDEAUX