Computer Vision

Introduction

Bassam Kurdy Ph.D

<bassam.kurdy@apinum.fr>

La *Vision par ordinateur CV - Quésaco ?*

- La **computer vision** désigne une technique **d'intelligence artificielle** permettant d'analyser des images captées par un équipement tel qu'une **caméra**.
- Concrètement, la computer vision se présente comme un outil basé sur l'IA capable de **reconnaître** une image, de la **comprendre**, et de **traiter** les informations qui en découlent.
- Pour beaucoup, la vision par ordinateur est l'équivalent, en termes d'IA, des **yeux humains** et de la capacité de notre **cerveau** à **traiter** et **analyser** les images perçues.
- La reproduction de la vision humaine par des ordinateurs constitue d'ailleurs l'un des grands objectifs de la computer vision.

Deep Learning & Computer Vision

« We should stop training radiologists now… in five years deeplearning is going to do better. Radiologists are like the coyote already over the edge of the cliff who hasn't yet looked down »

<< Il faut arrêter de former les radiologues maintenant… dans cinq ans le Deep Learning ira mieux. Les radiologues sont comme le coyote déjà au bord de la falaise qui n'a pas encore regardé vers le bas >>

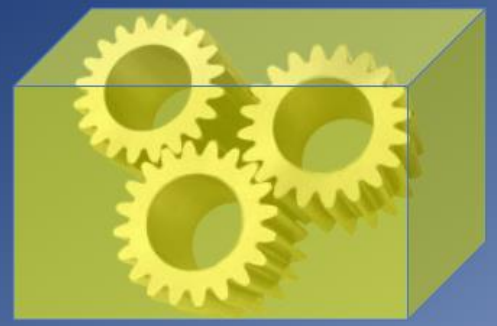
*Quelles sont les **applications** de la computer vision?*

On recense aujourd'hui de nombreux domaines d'application :

- **Les voitures autonomes**
- **Les systèmes de reconnaissance faciale**
- **L'industrie**
- **Assurance qualité**
- **La Santé (Diagnostic médical)**
- **Reconnaissance de caractères (OCR)**
- **Gérer les archives d'images**
- **Trademark et de copyright**
- **Problèmes de sécurité**
- **Problèmes militaires**
- **SIG et télédétection**
- **… etc**

Machine Learning

= Classifieur



Chat

Lapin

Machine Learning

= Classifieur

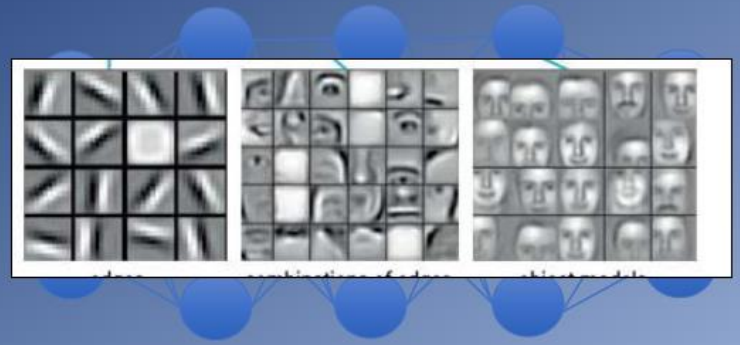
Entrainement

Machine Learning

- Contours
- Densité
- Texture

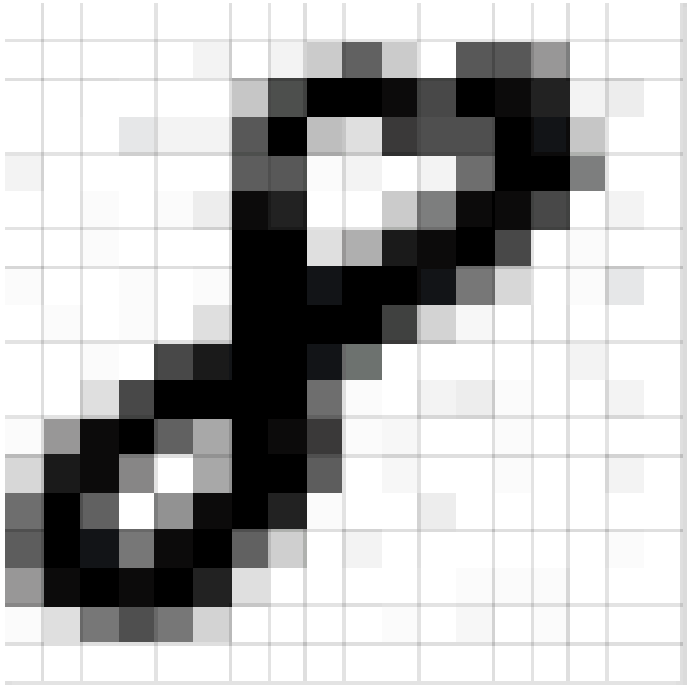
Chat

Deep Neural Network



Chat

Une image est une matrice de valeurs de pixels



Pourquoi utiliser OpenCV ?

- Les algorithmes *d'**OpenCV*** permettent d'appliquer divers traitements sur les images pour faciliter la détection d'éléments précis dans celles-ci.
- Voici quelques exemples :
  - **Le Thresholding (seuillage)** d'une image permet de définir une valeur de pixel (correspondant à une couleur ou niveau de gris) qui servira de seuil. Au-dessus ou en dessous de cette valeur (selon l'algorithme), tous les pixels se verront assigner une autre valeur.
  - **Les filtres de détection de contours**
  - **Les filtres de lissage**
  - **Les filtres morphologiques**
  - **… etc.**

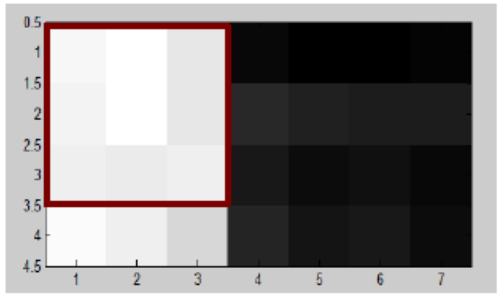
Exemple « convolution »

Appellation de Filtre (*Filter*), ou Noyau (*Kernel*)

Soit une image $I \in \mathbb{R}^{m \times n}$ et un filtre de convolution $F \in \mathbb{R}^{k \times k}$ de taille impaire $k = 2d + 1$ (avec $d \in \mathbb{N}^+$):

$$(I * F)[x, y] = \sum_{i=-d}^{+d} \sum_{j=-d}^{+d} I[x+i, y+j] \times F[i+d+1, j+d+1]$$

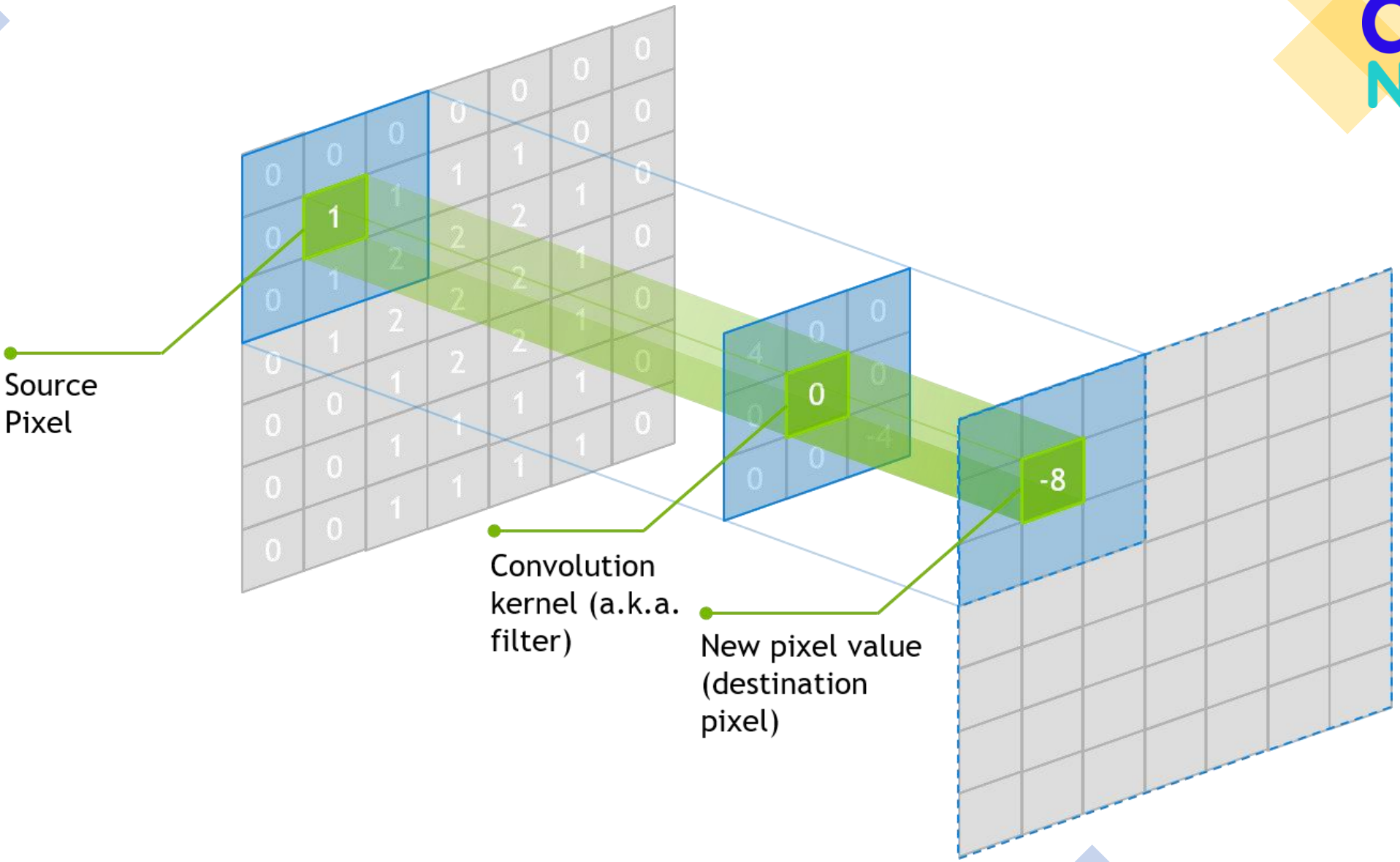
$$F = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$



207	210	195	63	57	56	59
204	212	197	82	76	74	75
202	198	202	72	65	67	63
209	201	187	78	69	71	64

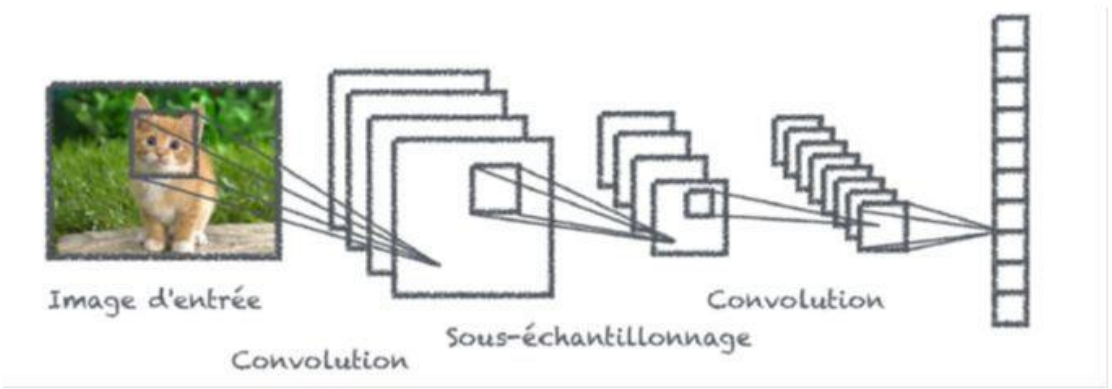
$I * F$

	-26	-533	-517	-28	
	-29	-505	-513	-25	


Source Pixel
Convolution kernel (a.k.a. filter)
New pixel value (destination pixel)
1
0
-8

Les réseaux de neurones convolutionnels

En apprentissage automatique, un **réseau de neurones convolutifs** ou **réseau de neurones à convolution** (en anglais ***CNN*** ou ***ConvNet*** pour *Convolutional Neural Networks*) est un type de réseau de neurones artificiels acycliques (*feed-forward*), dans lequel le motif de connexion entre les neurones est inspiré par le cortex visuel des animaux.

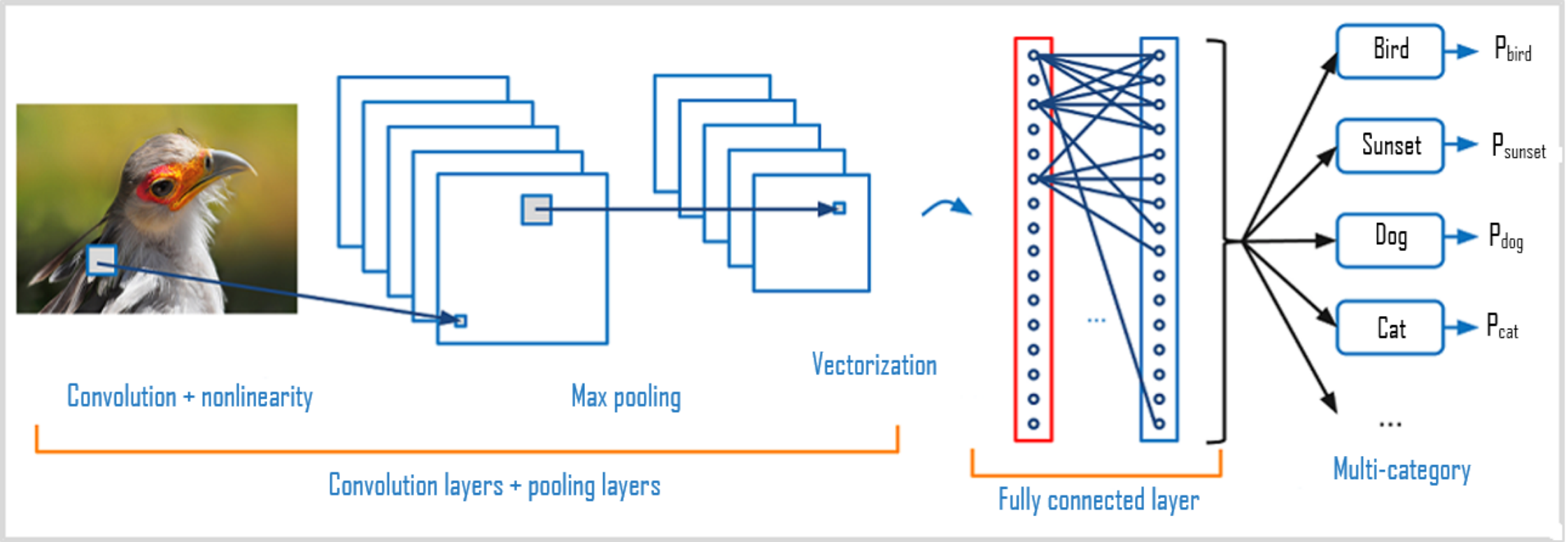


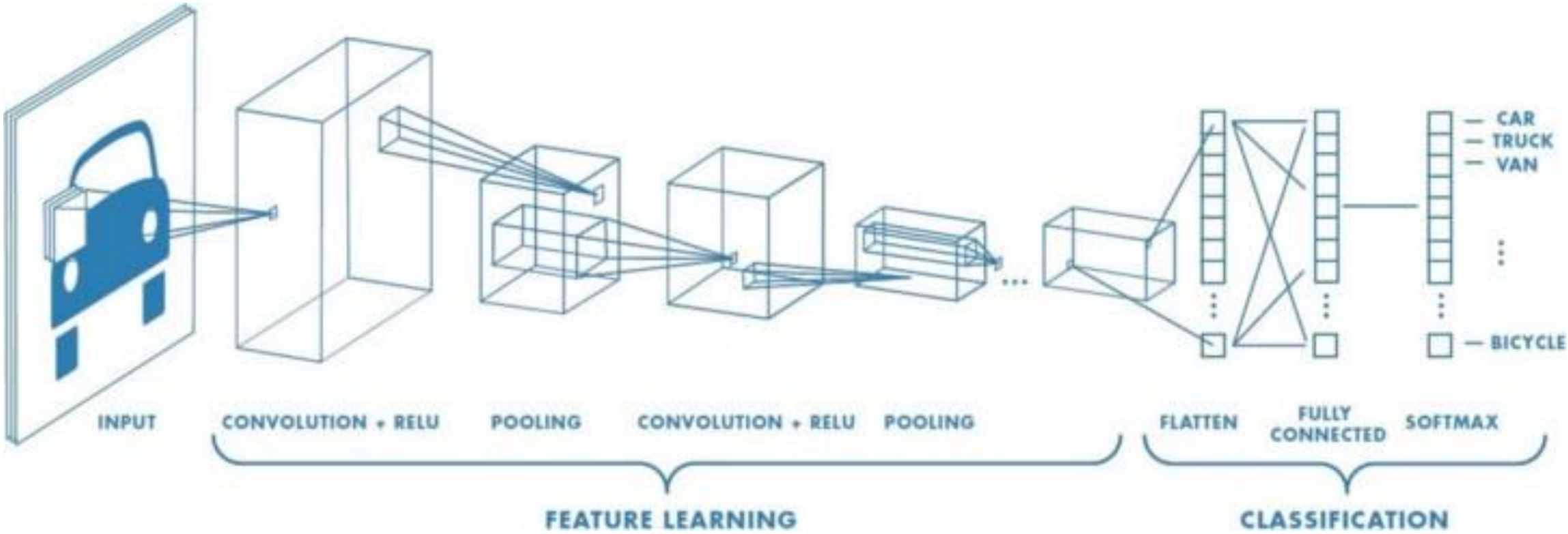
33

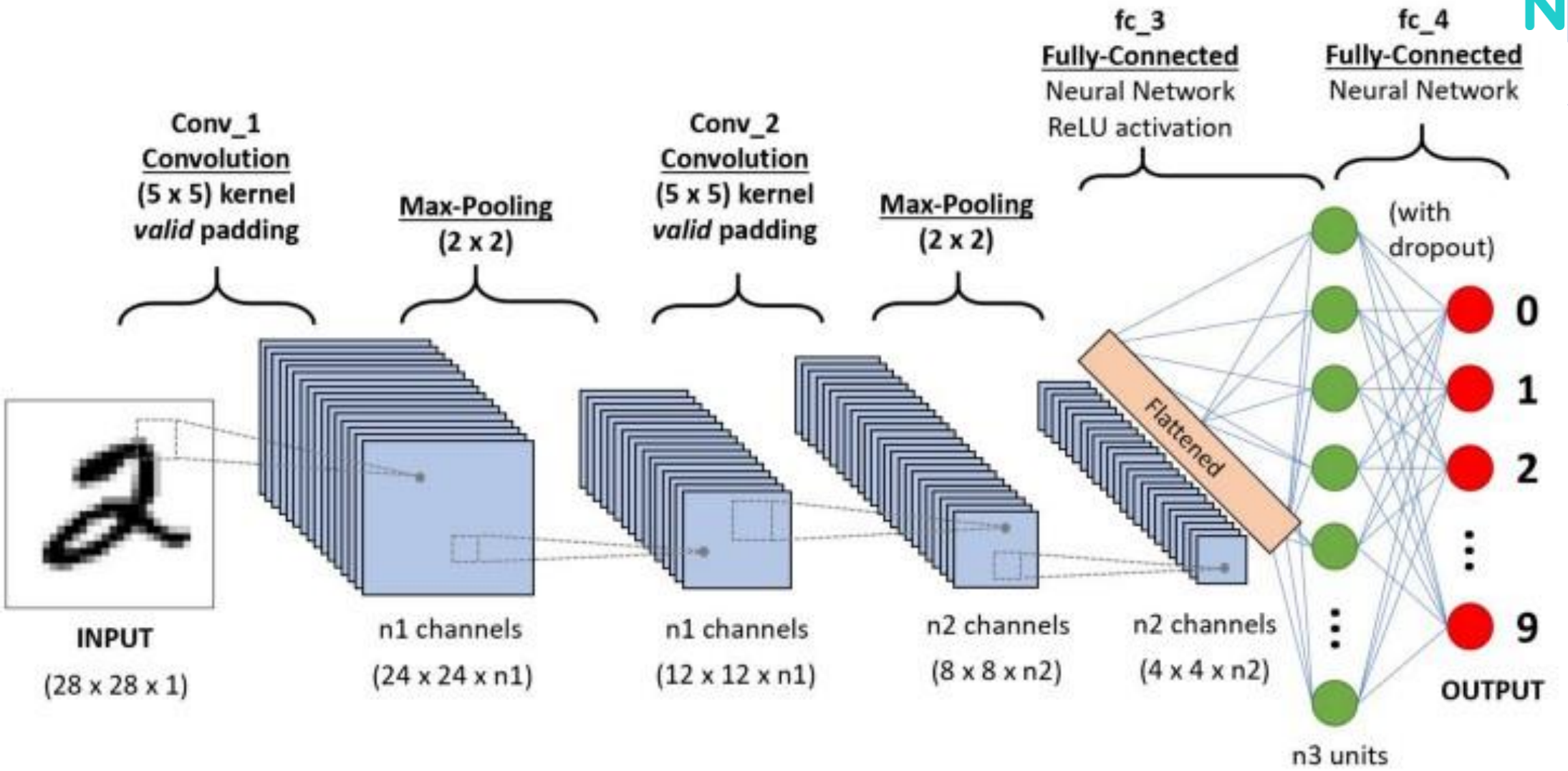
Les réseaux de neurones convolutifs (CNN)

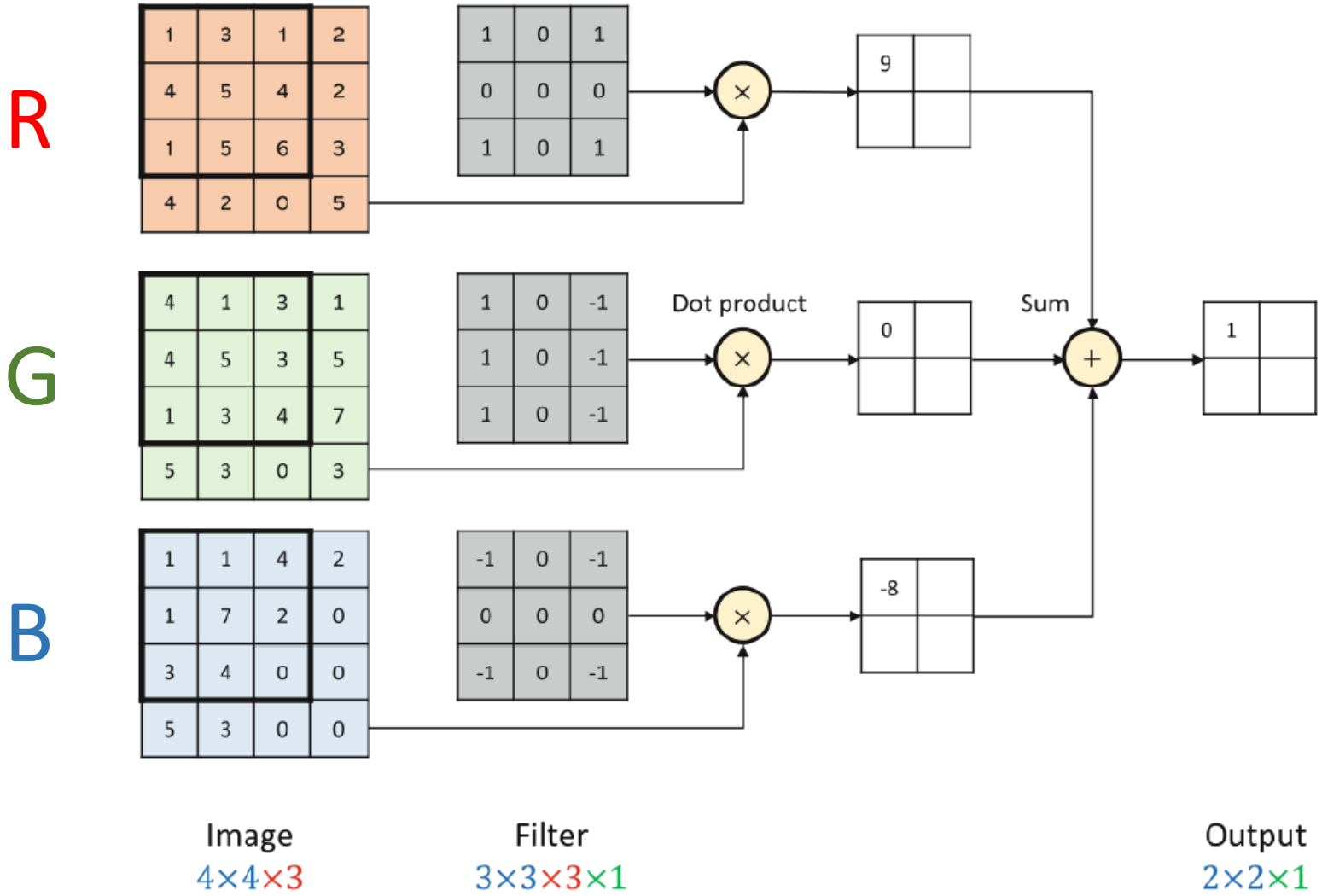
- Motivations sur les images :
  - Corrélation locale des données
  - Détection de formes
  - Invariance par translation : les formes peuvent se trouver à différents endroits de l'image

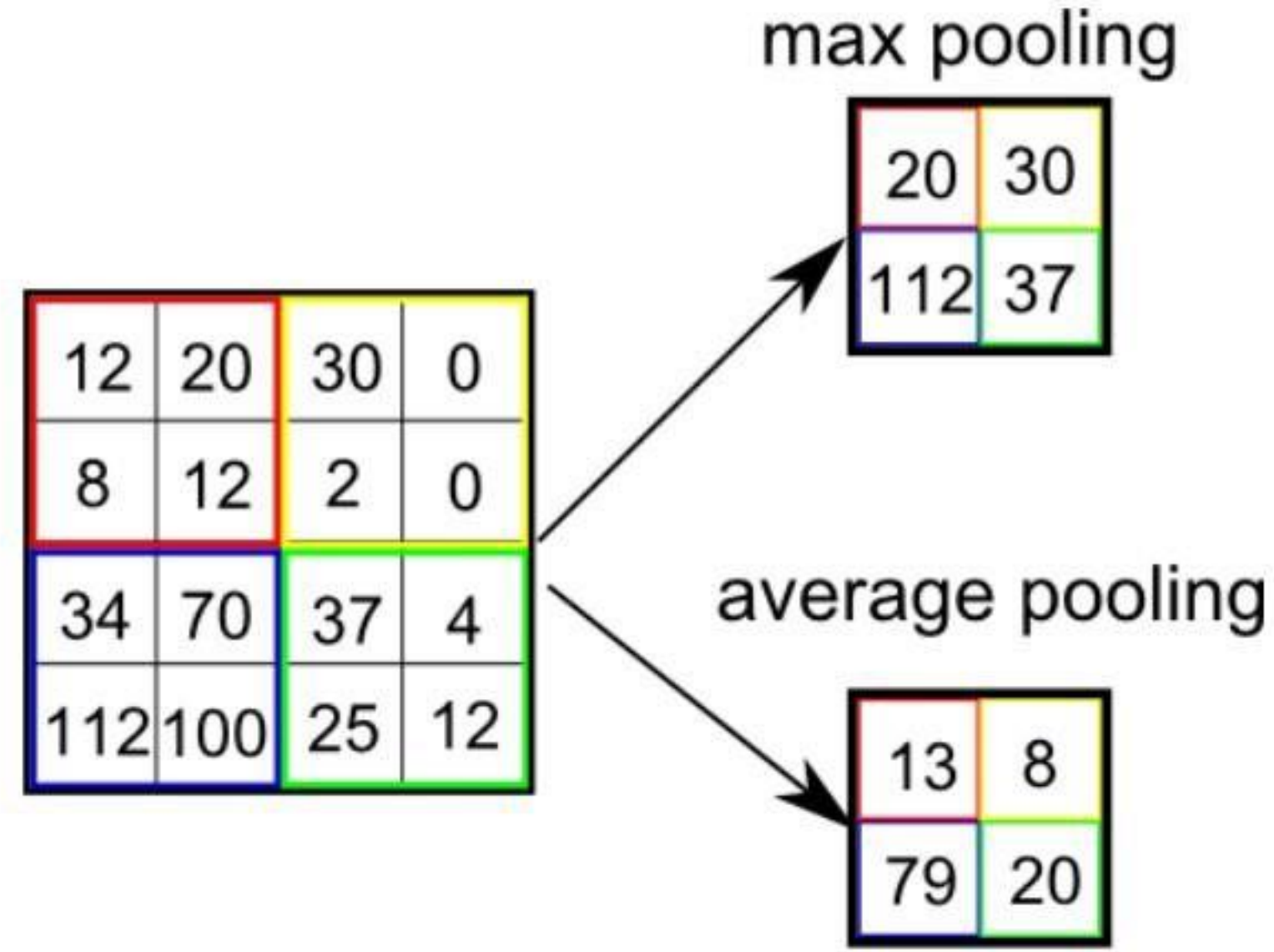
- La connaissance du problème influence la conception de l'architecture :
  - Les images possède une structure 2D
  - Forte corrélation locale dans les valeurs des pixels


Bird
Sunset
Dog
Cat
...
P_{bird}
P_{sunset}
P_{dog}
P_{cat}
Vectorization
Convolution + nonlinearity
Max pooling
Convolution layers + pooling layers
Fully connected layer
Multi-category


CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
FEATURE LEARNING
CLASSIFICATION


Conv_1
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
Conv_2
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
fc_3
Fully-Connected
Neural Network
ReLU activation
fc_4
Fully-Connected
Neural Network
(with
dropout)
Flattened
INPUT
(28 x 28 x 1)
n1 channels
(24 x 24 x n1)
n1 channels
(12 x 12 x n1)
n2 channels
(8 x 8 x n2)
n2 channels
(4 x 4 x n2)
n3 units
0
1
2
9
OUTPUT


R
G
B
1 3 1 2
4 5 4 2
1 5 6 3
4 2 0 5
4 1 3 1
4 5 3 5
1 3 4 7
5 3 0 3
1 1 4 2
1 7 2 0
3 4 0 0
5 3 0 0
1 0 1
0 0 0
1 0 1
1 0 -1
1 0 -1
1 0 -1
-1 0 -1
0 0 0
-1 0 -1
Dot product
9
0
-8
Sum
1
Image
4×4×3
Filter
3×3×3×1
Output
2×2×1


max pooling
12 20 30 0
8 12 2 0
34 70 37 4
112 100 25 12
20 30
112 37
average pooling
13 8
79 20

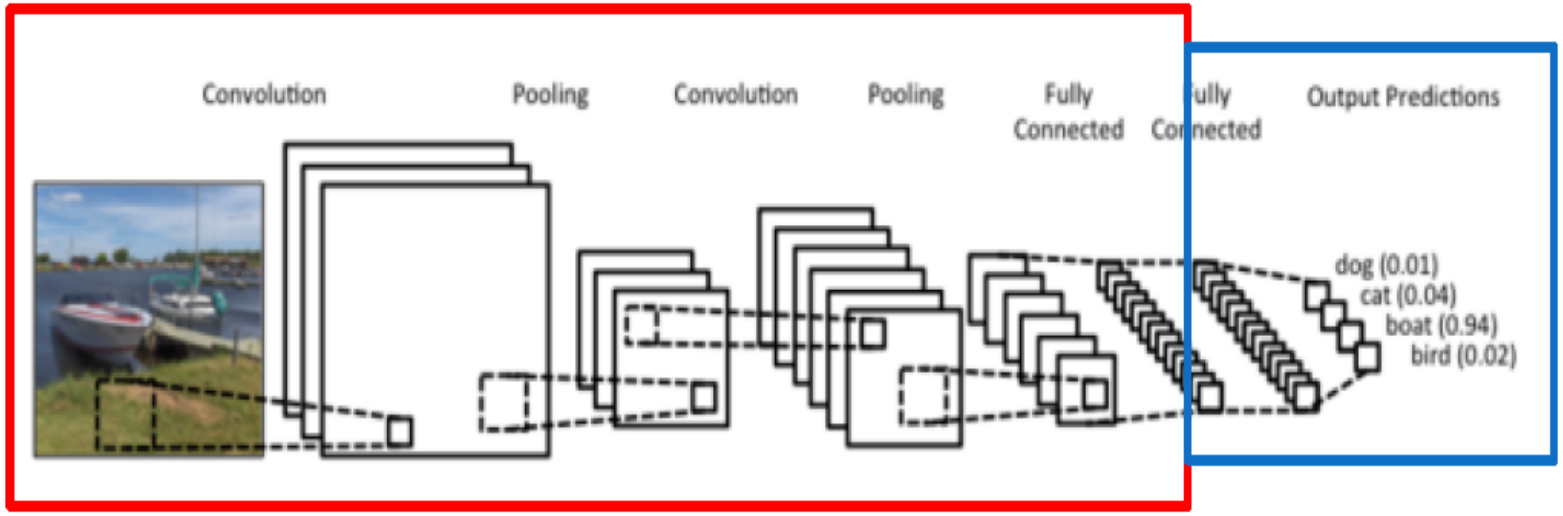
Les réseaux de neurones convolutifs (CNN)

- Sous Keras (en **gras : ce qui est nouveau,** *en italique, ce qui est modifiable*) :
- keras.layers.Conv2D(***filters*, *kernel_size*, strides=*(1, 1)*, padding='*valid*',** activation=*None*, use_bias=*True*, kernel_initializer='*glorot_uniform*', bias_initializer='*zeros*', kernel_regularizer=*None*, bias_regularizer=*None*)
- keras.layers.MaxPooling2D(**pool_size=*(2, 2)*, strides=*None*, padding='*valid*'**)
- keras.layers.UpSampling2D(**size=*(2, 2),* interpolation='*nearest*'**) : opération inverse du pooling
- Ces trois couches existent aussi en version 1D et 3D
- keras.layers.Flatten() (pour mettre les données sur une seule ligne avant des fully-connected layers)
- Conv2D layer (keras.io)

Transfer Learning

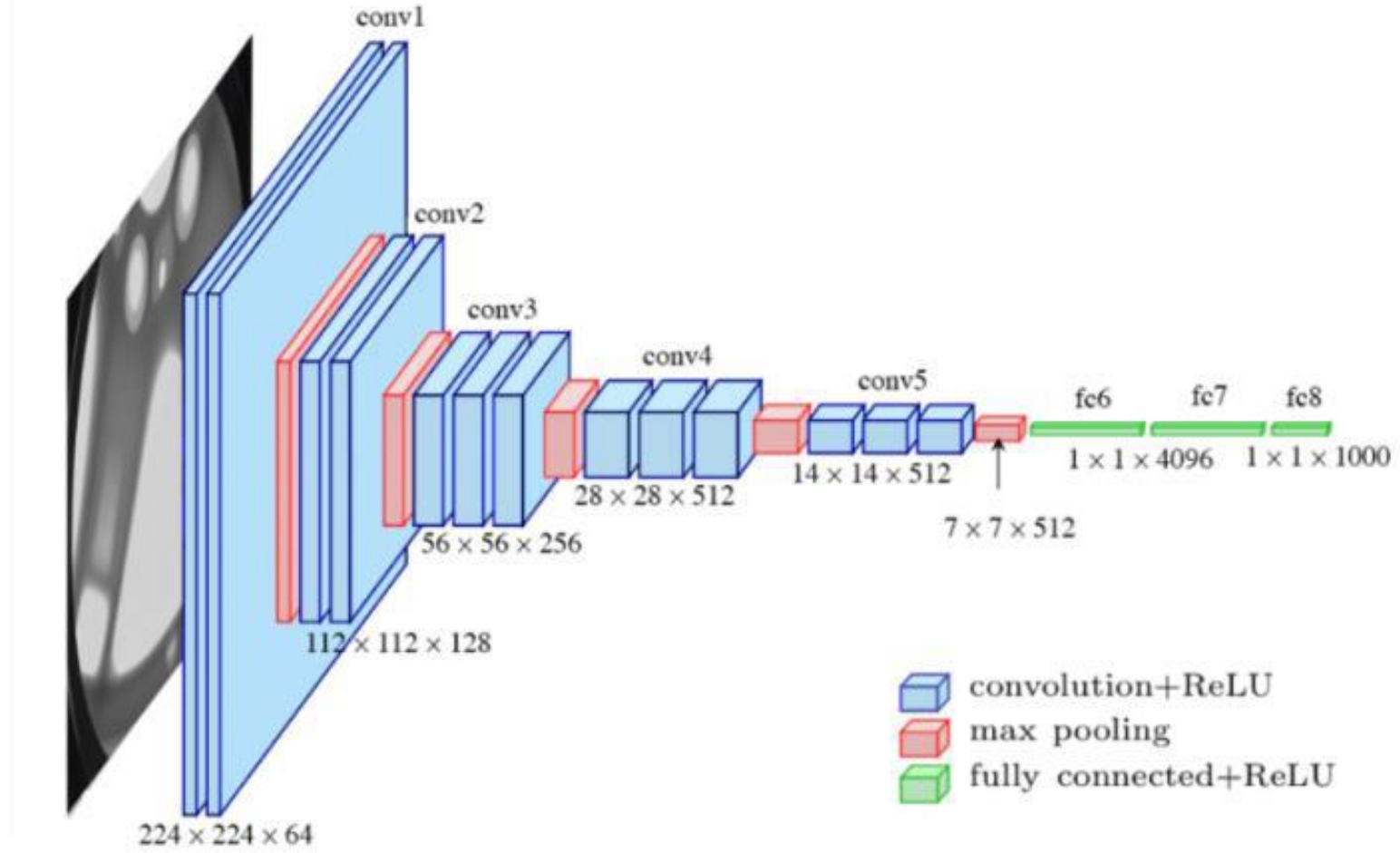
- Idée : conserver l'extraction des caractéristiques apprise sur d'autres problématiques
  - Revient à conserver des couches de convolution apprises sur un problème similaire
  - On ne change que la (ou éventuellement les) dernière(s) couche(s) d'identification

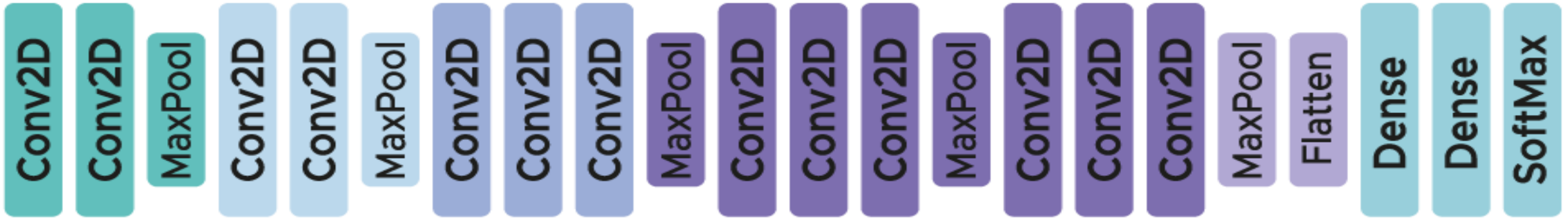
Fixée (déjà apprise)



On apprend seulement ces couches

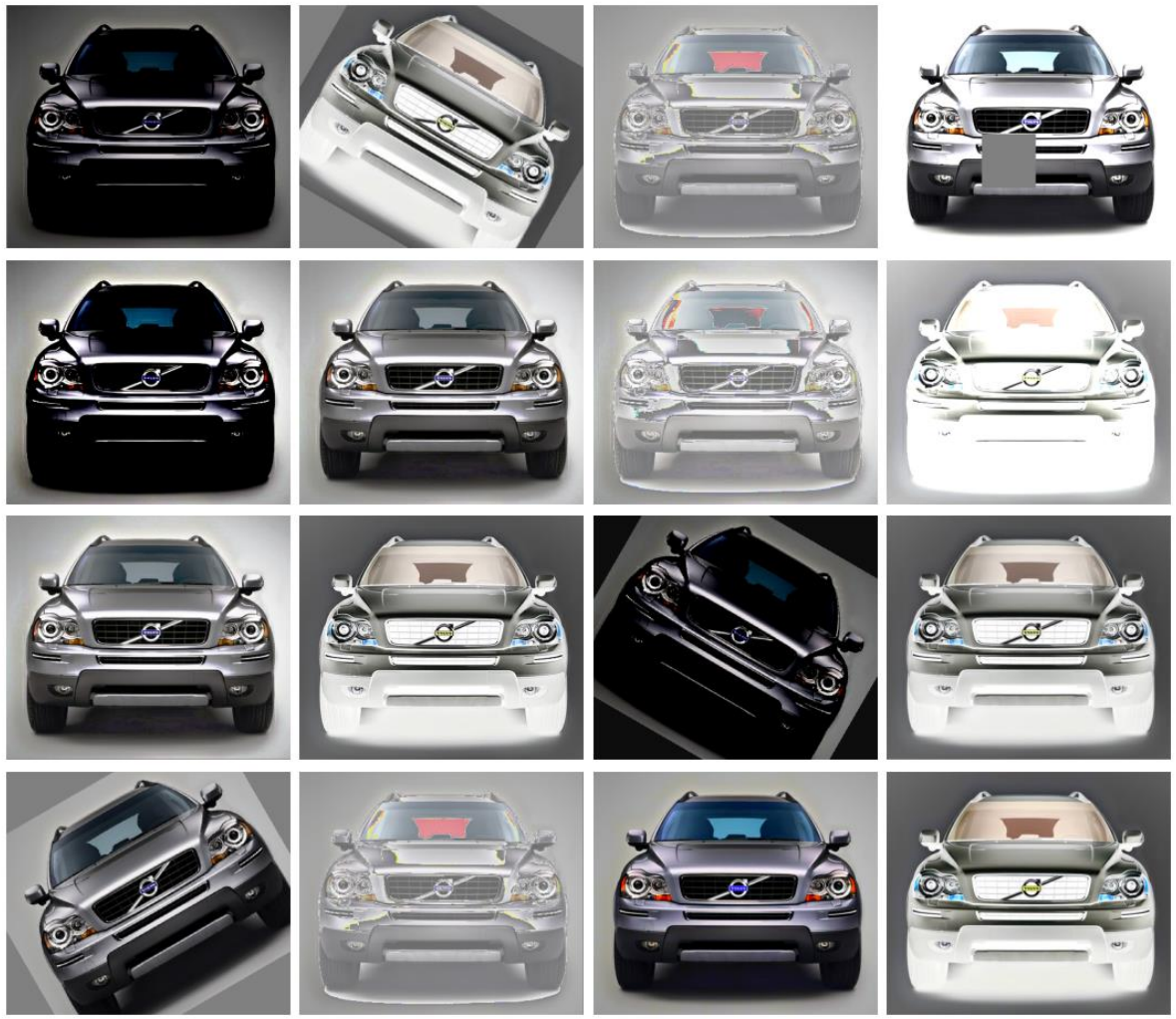
VGG16


conv1
conv2
conv3
conv4
conv5
fc6
fc7
fc8
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU



VGG16 Architecture

Data Augmentation



Data Augmentation

```
from keras.applications.vgg16 import preprocess_input
from keras.preprocessing.image import ImageDataGenerator

train_data_generator = ImageDataGenerator(
        Preprocessing_function = preprocess_input,
        # data augmentation
        rotation_range = 10,
        width_shift_range = 0.1,
        height_shift_range = 0.1,
        zoom_range = 1.1,
        horizontal_flip = True
        )

test_data_generator = ImageDataGenerator(
preprocessing_function = preprocess_input)
```