

Computer vision – Part 2

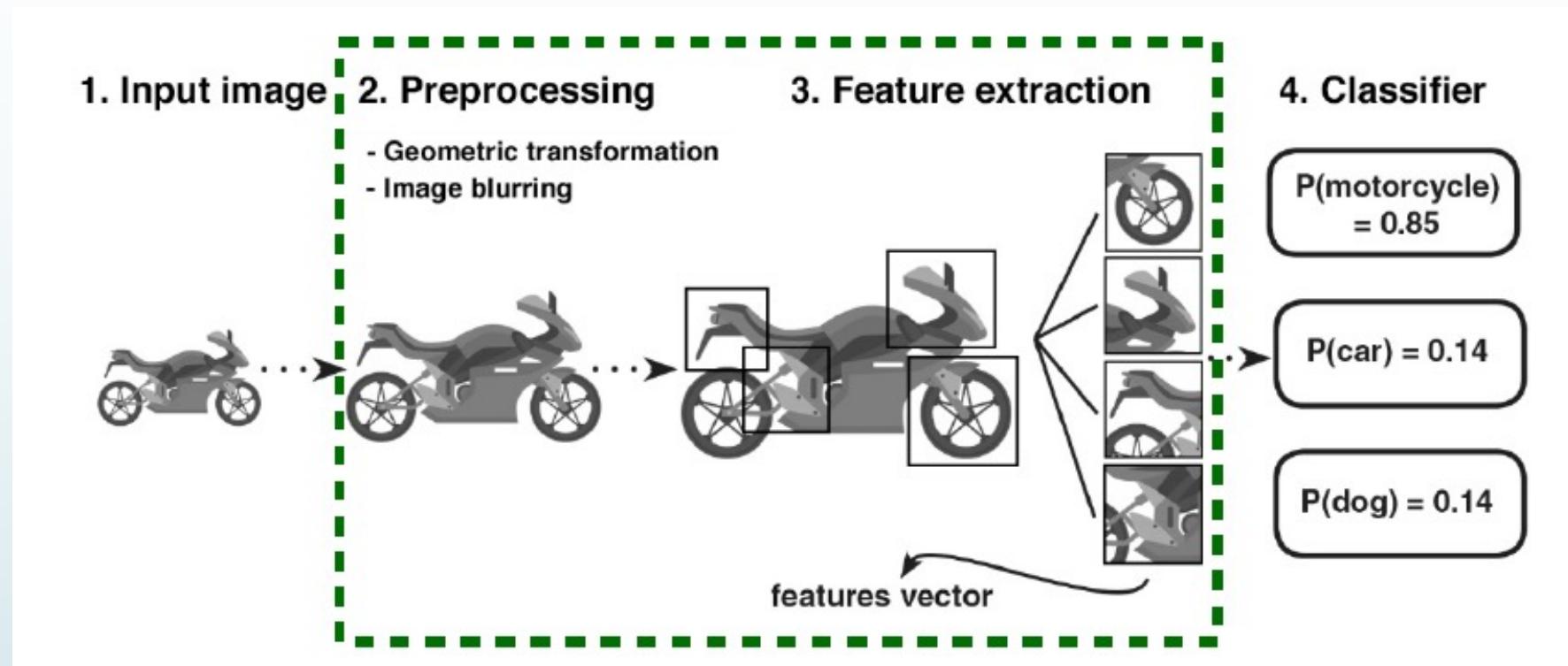
Cours 15, IFT 6758

Automne 2024

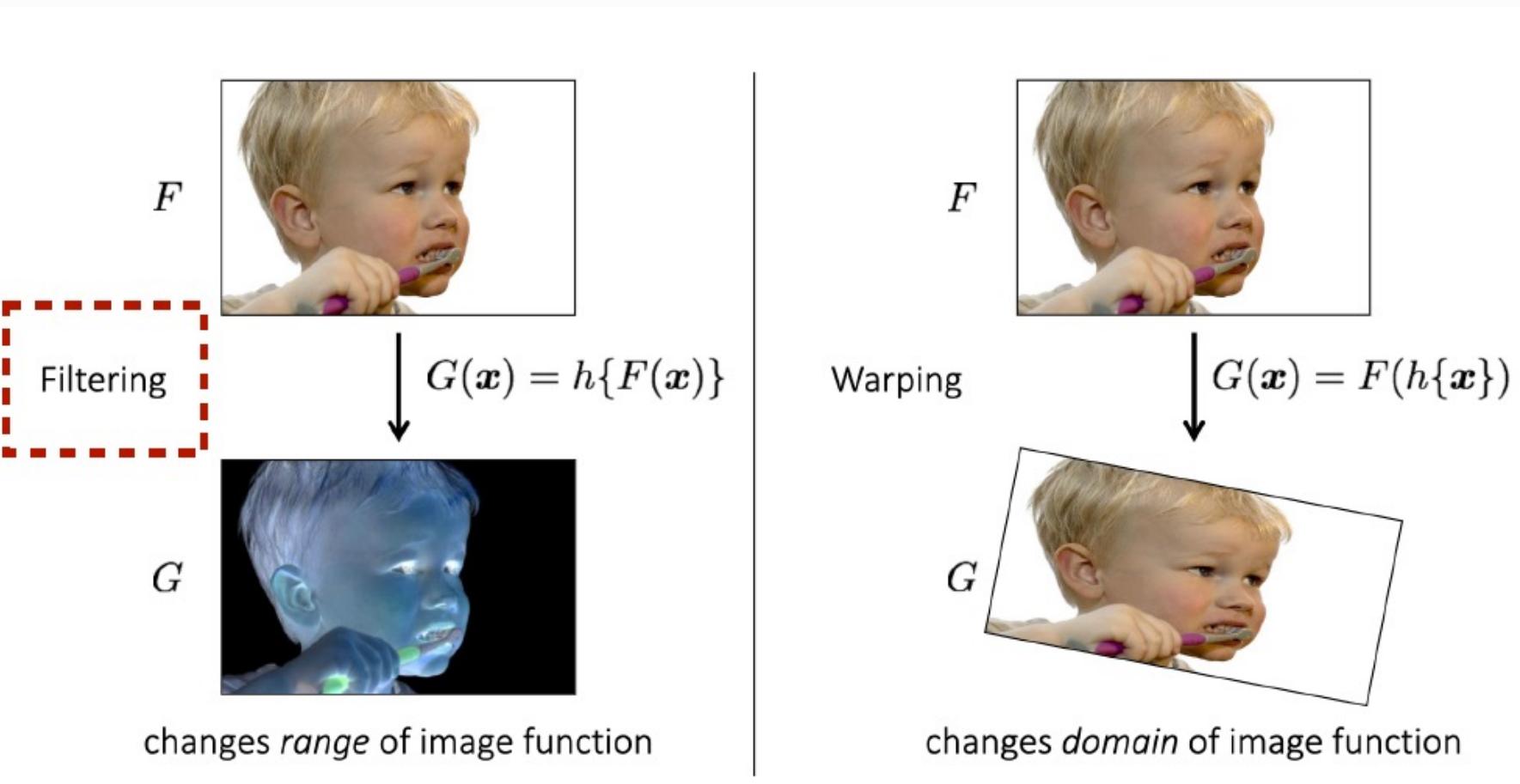
Aperçu

- ▶ Tâches en CV (vision par ordinateur)
- ▶ Les défis du CV
- ▶ Le pipeline CV
- ▶ Image d'entrée:
 - ▶ **Prétraitement**
 - ▶ **Extraction de fonctionnalités**
 - ▶ **Prédiction**

Pipeline de vision par ordinateur

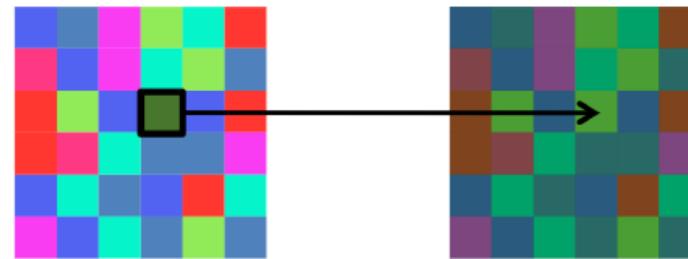


Transformations d'image



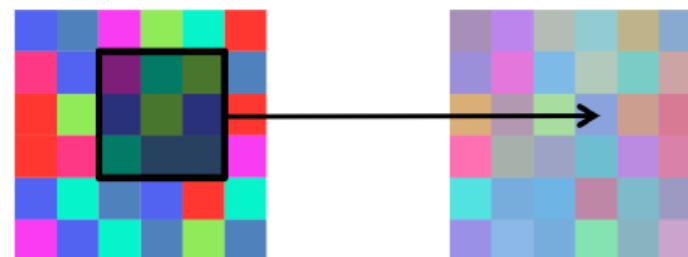
Filtrage

Point Operation



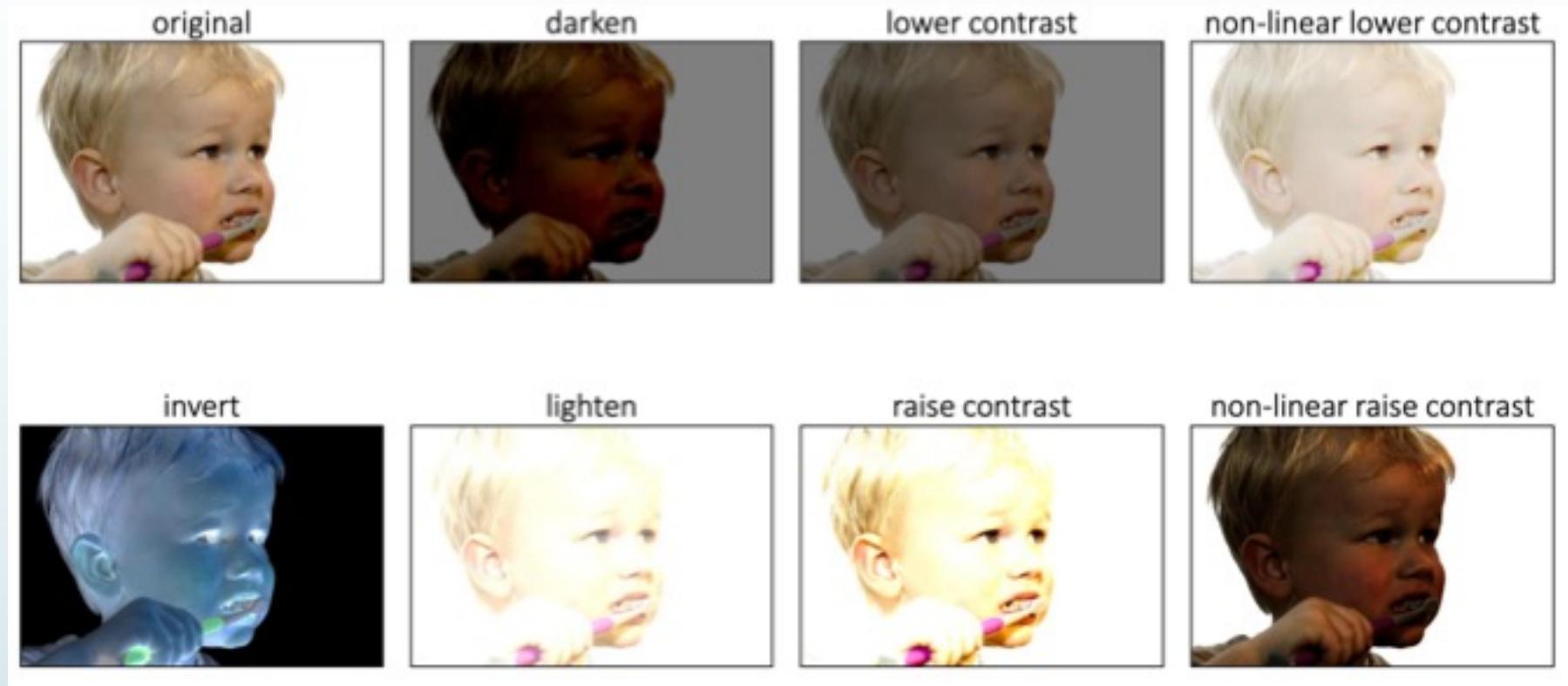
1D
point processing

Neighborhood Operation

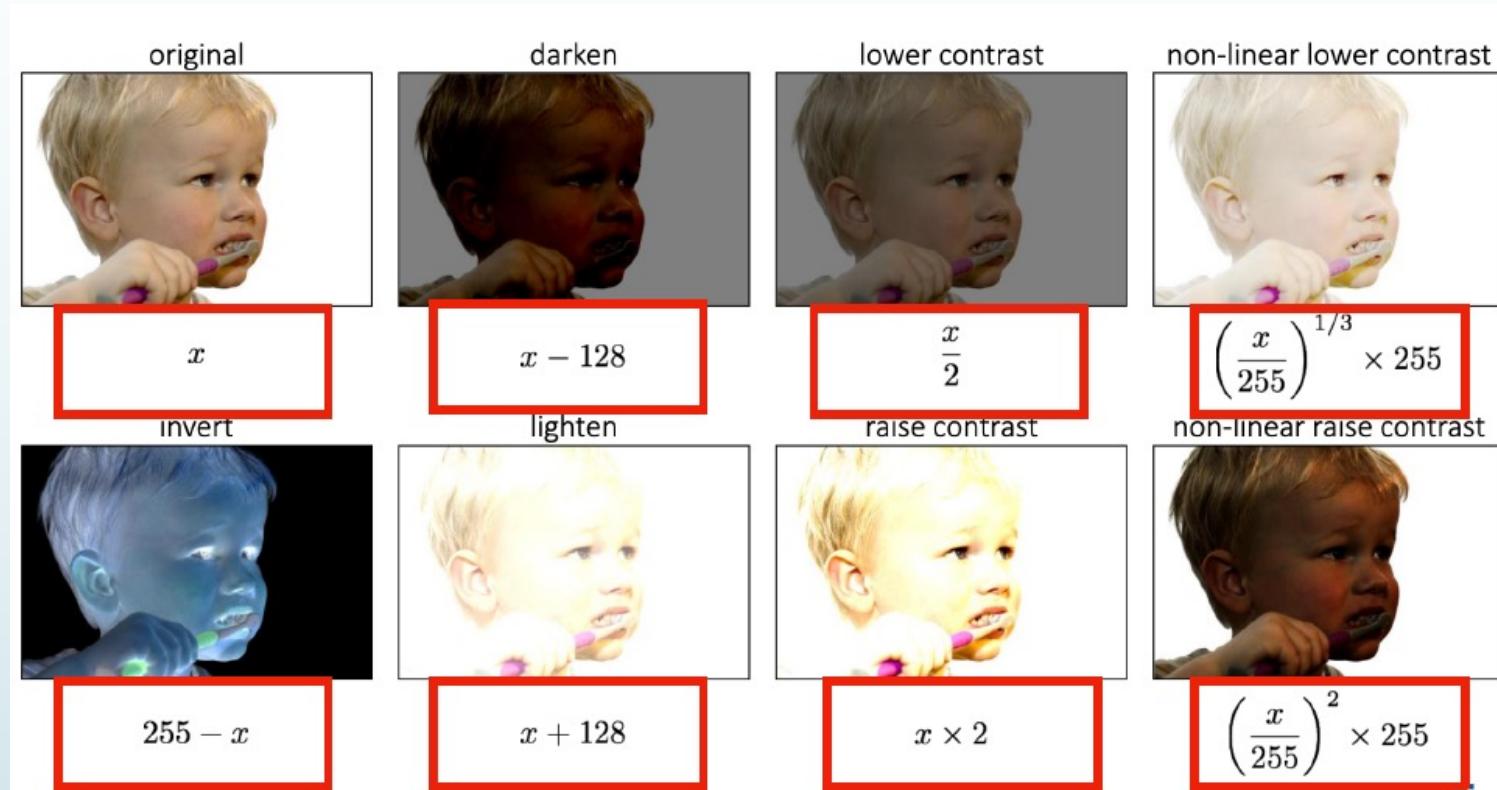


2D
“filtering”

Traitements point par point



Traitement point par point (Comment les mettre en œuvre ?)



Filtrage

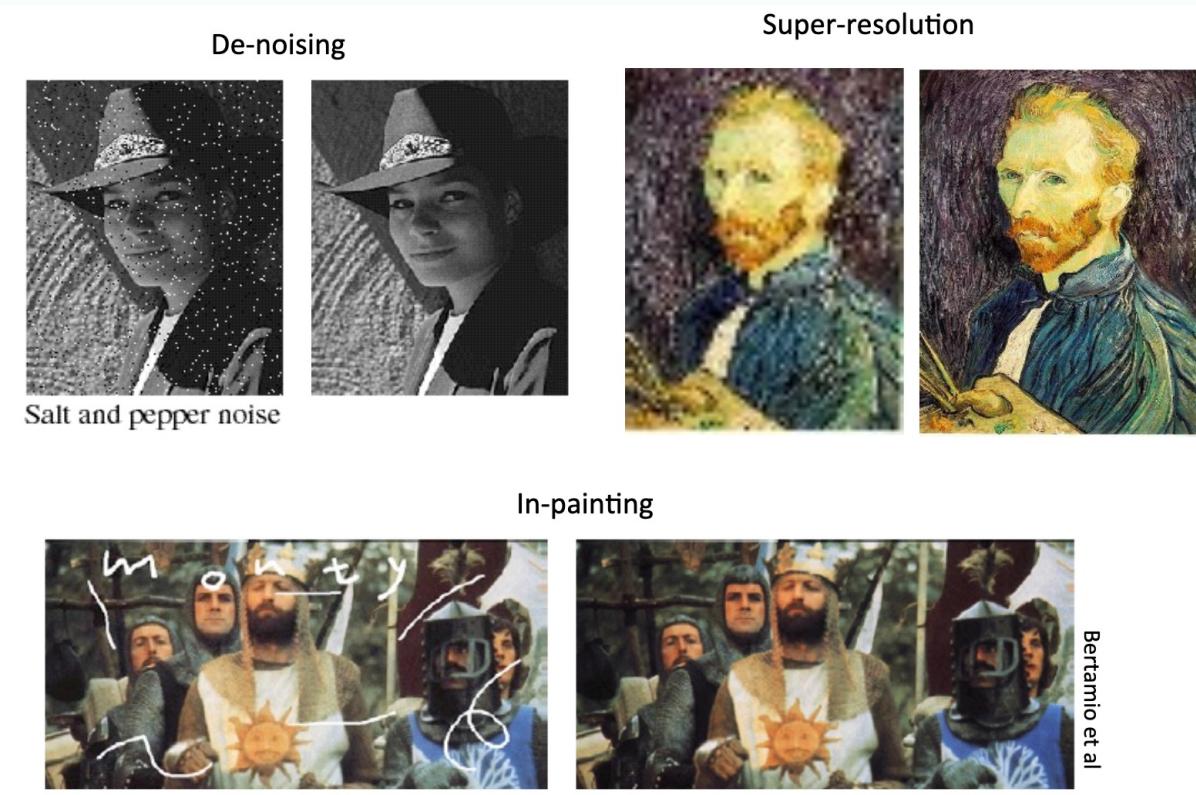
► **Filtrage:**

- Forme une nouvelle image dont les pixels sont une combinaison des valeurs originales

► **BUT:**

- Extraire des informations utiles de l'image (lignes, coins, morceaux)
- Modifier ou améliorer les propriétés
 - (suppression du bruit, super-résolution)

Exemples d'amélioration



Filtrage local

Modifie les pixels avec une fonction qui dépend des pixels voisins.

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

10	5	3
4	5	1
1	1	7

Local image data

Some function



		7

Modified image data

Formalisation d'un filtre

$$f[n, m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n, m]$$

$$g = \mathcal{S}[f], \quad g[n, m] = \mathcal{S}\{f[n, m]\}$$

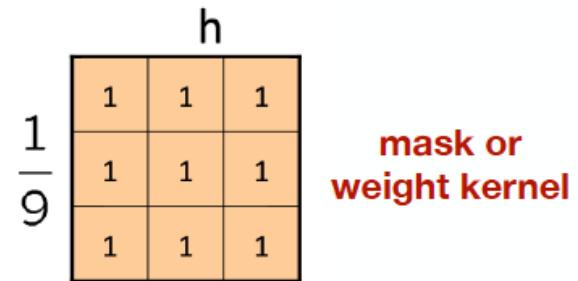
$$f[n, m] \xrightarrow{\mathcal{S}} g[n, m]$$

Exemple de filtre : Moyenne mobile

- Également connu sous le nom **de filtre boîte**

-

$$\begin{aligned}g[n, m] &= \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k, l] \\&= \frac{1}{9} \sum_{k=-1}^1 \sum_{l=-1}^1 f[n-k, m-l]\end{aligned}$$



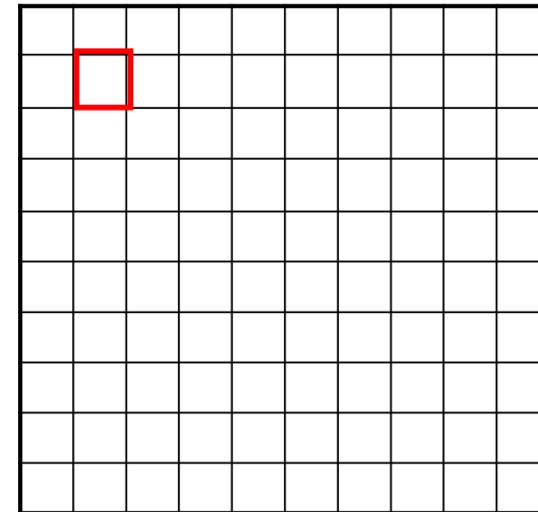
$$(f * h)[m, n] = \frac{1}{9} \sum_{k,l} f[k, l] h[m - k, n - l]$$

- Moyenne mobile 2D sur une fenêtre de voisinage 3x3

Exemple de filtre : Moyenne mobile

$$F[x, y]$$

$$G[x, y]$$



$$(f * h)[m, n] = \sum_{k,l} f[k, l] h[m - k, n - l]$$

Filter example: Moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

$G[x, y]$

0	10										

$$(f * h)[m,n] = \sum_{k,l} f[k,l] h[m-k,n-l]$$

Exemple de filtre : Moyenne mobile

$F[x, y]$

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	90	90	0	0
0	0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

$G[x, y]$

			0	10	20						

$$(f * h)[m, n] = \sum_{k,l} f[k, l] h[m - k, n - l]$$

Exemple de filtre : Moyenne mobile

$F[x, y]$

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$G[x, y]$

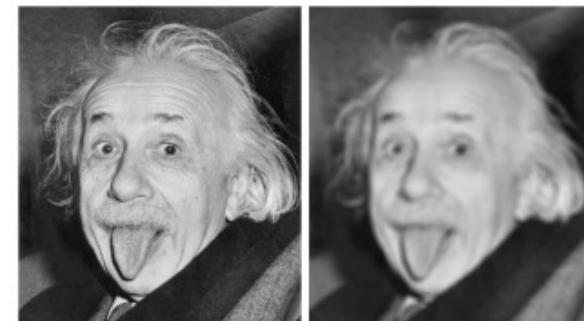
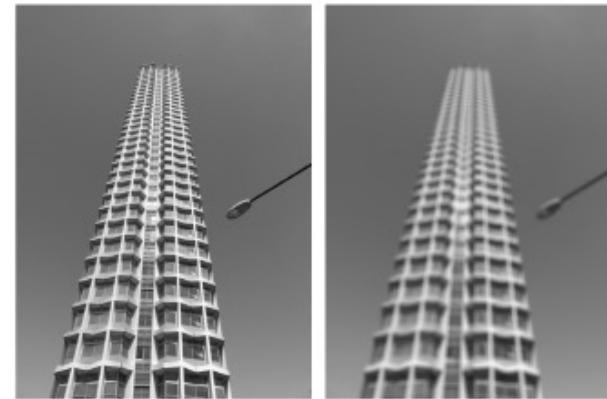
	0	10	20	30	30	30	20	10		
	0	20	40	60	60	60	40	20		
	0	30	60	90	90	90	60	30		
	0	30	50	80	80	90	60	30		
	0	30	50	80	80	90	60	30		
	0	20	30	50	50	60	40	20		
	10	20	30	30	30	30	20	10		
	10	10	10	0	0	0	0	0		

$$(f * h)[m, n] = \sum_{k,l} f[k, l] h[m - k, n - l]$$

Source: S. Seitz

CONVOLUTION

Exemple de filtre : Moyenne mobile



Achieve smoothing effect (remove sharp features)

Rôle de lissage (enlève les contours prononcés)

Exemple de filtre non linéaire : segmentation d'image

Segmentation utilisant un seuil:

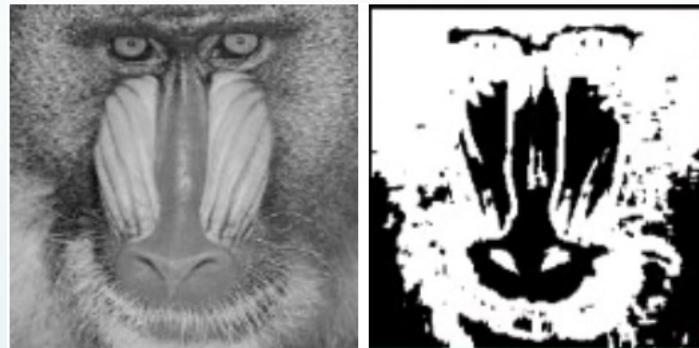
- Image segmentation based on a simple threshold:

$$g[n, m] = \begin{cases} 255, & f[n, m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

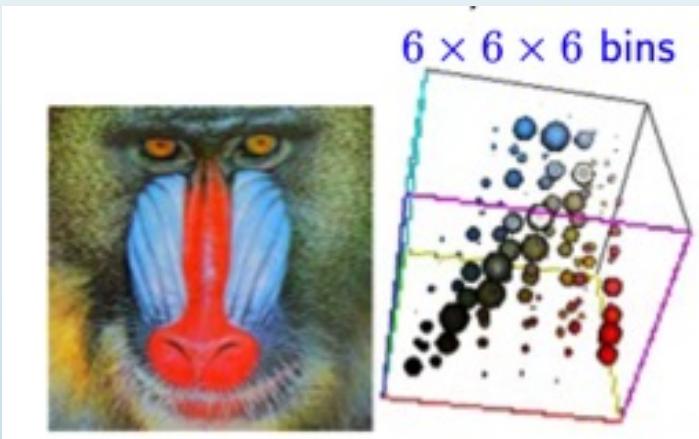


Exemple de filtre : segmentation d'image

- ▶ **Non contextuel** : regroupement de pixels ayant des caractéristiques globales similaires.



- ▶ **Contextuel** : regroupement de pixels avec des caractéristiques similaires et dans des emplacements proches.



Exemple de filtre : Moyenne mobile

$F[x, y]$

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30	30	30	20	10		
	0	20	40	60	60	60	40	20		
	0	30	60	90	90	90	60	30		
	0	30	50	80	80	90	60	30		
	0	30	50	80	80	90	60	30		
	0	20	30	50	50	60	40	20		
	10	20	30	30	30	30	20	10		
	10	10	10	0	0	0	0	0		
	0	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0	0		

$$(f * h)[m, n] = \sum_{k,l} f[k, l] h[m - k, n - l]$$

Source: S. Seitz

CONVOLUTION

Convolution properties

Commutative: $f * g = g * f$

Associative: $(f * g) * h = f * (g * h)$

Distributive: $(f + g) * h = f * h + g * h$

Linear: $(af + bg) * h = af * h + bg * h$

Shift Invariant: $f(x+t) * h = (f * h)(x+t)$

Differentiation rule:

$$\frac{\partial}{\partial x}(f * g) = \frac{\partial f}{\partial x} * g$$

Equivariance par translation

- Le filtre remplace chaque pixel par une combinaison linéaire de ses voisins (et éventuellement de lui-même). La combinaison est déterminée par le noyau du filtre.

If $f[n, m] \xrightarrow{\mathcal{S}} g[n, m]$ then

$$f[n - n_0, m - m_0] \xrightarrow{\mathcal{S}} g[n - n_0, m - m_0]$$

for every input image $f[n, m]$ and shifts n_0, m_0

Equivariance par translation

- Le décalage du filtre de moyenne mobile est-il equivariant ?

$$F[x, y]$$

$$G[x, y]$$

Equivariance par translation

- Le décalage du système de moyenne mobile est-il equivariant ?

$$f[n, m] \xrightarrow{\mathcal{S}} g[n, m] = \frac{1}{9} \sum_{k=-1}^1 \sum_{l=-1}^1 f[n - k, m - l]$$

$$f[n - n_0, m - m_0]$$

$$\xrightarrow{\mathcal{S}} g[n, m] = \frac{1}{9} \sum_{k=-1}^1 \sum_{l=-1}^1 f[n - k, m - l]$$

$$= \frac{1}{9} \sum_{k=-1}^1 \sum_{l=-1}^1 f[(n - n_0) - k, (m - m_0) - l]$$

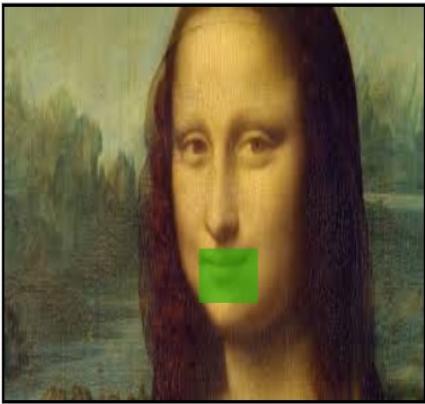
$$= g[n - n_0, m - m_0]$$

Yes!

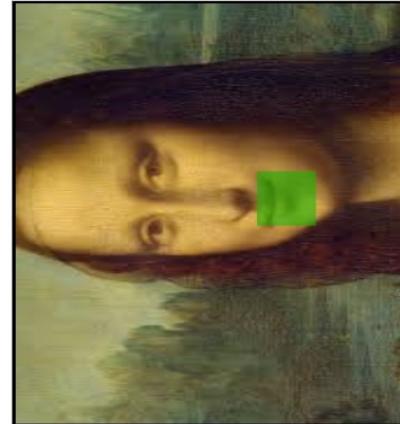
Invariance et Equivariance

- ▶ Entrée: x
- ▶ Sortie: $f(x)$
- ▶ f est **invariant** par translation si pour toute translation t on a
$$f(t(x)) = f(x)$$
- ▶ f est **équivariant** par translation si pour toute translation t on a
$$f(t(x)) = t(f(x))$$
- ▶ **Règle:** souvent on veut
 - ▶ des **caractéristiques**/représentation **équivariante** et
 - ▶ des **prédition invariantes**.

Example:



Rotation
→



Rotation Equivariance
in image features



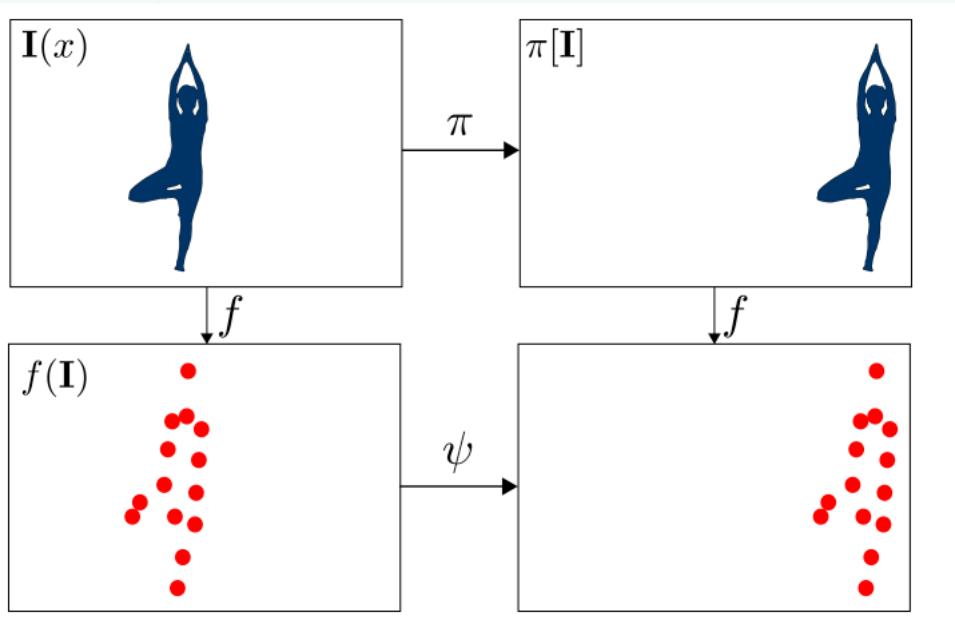
Rotation
→



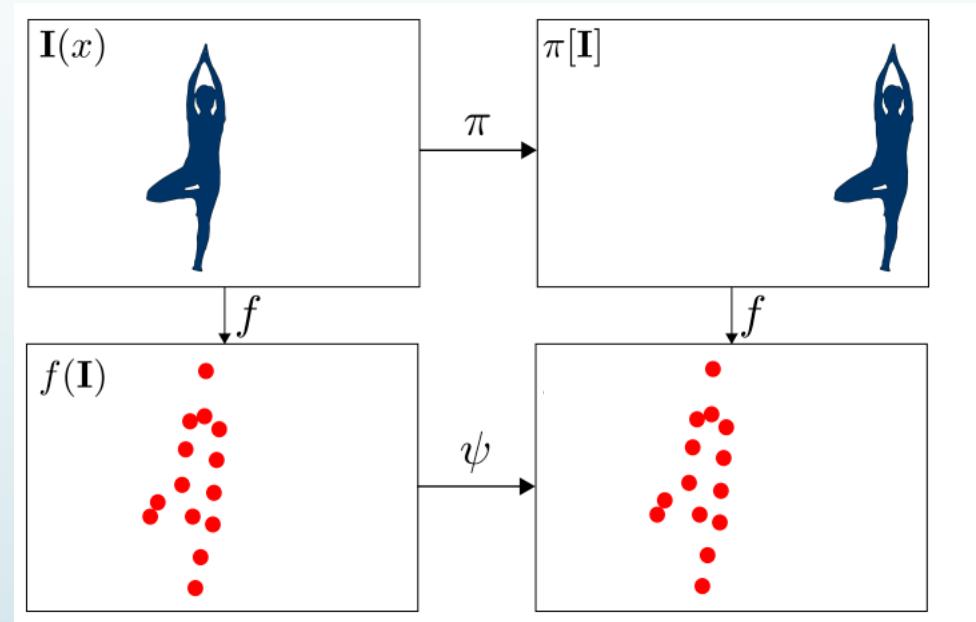
Rotation Equivariance
in image features

Exercice:

A)



B)



Filtrage Linéaire

- Le filtrage linéaire est une combinaison linéaire de valeurs de pixels voisins.

- S is a linear system (function) iff it *S satisfies***

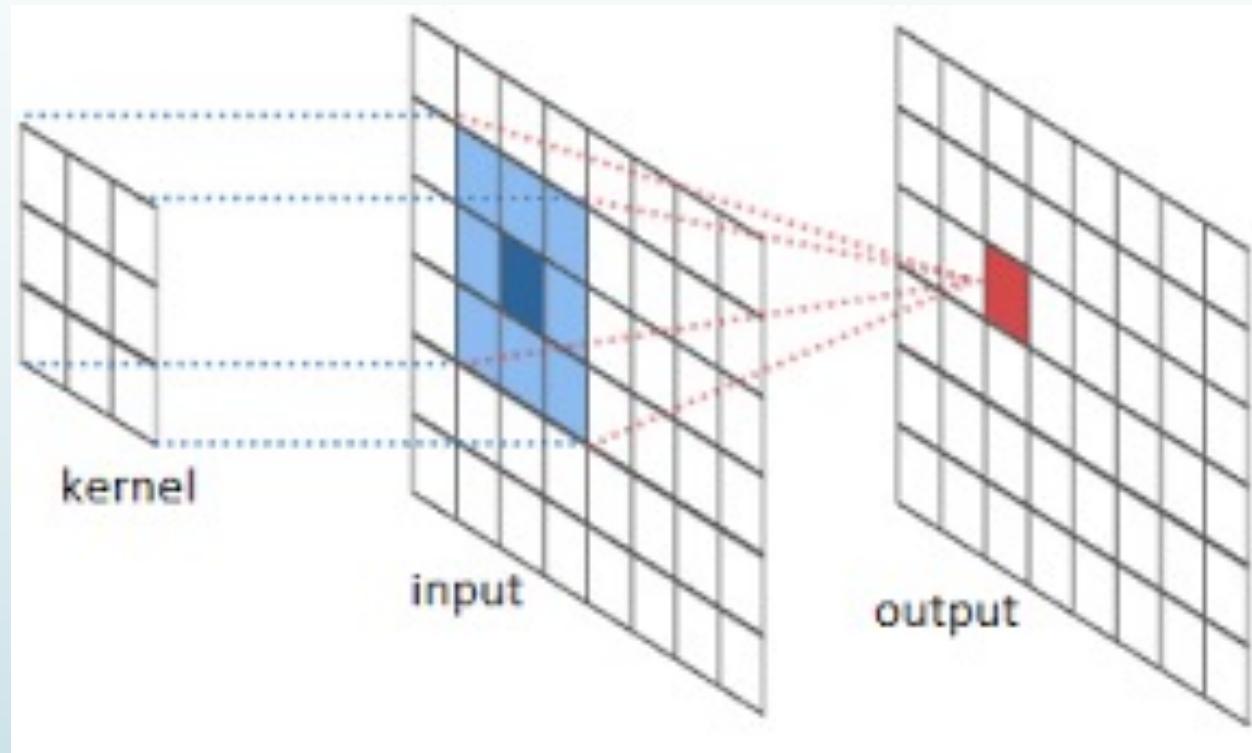
$$S[\alpha f_1 + \beta f_2] = \alpha S[f_1] + \beta S[f_2]$$

superposition property

- Is the moving average system a linear system?
- Is thresholding a linear system?

Convolution

- ▶ Tout opérateur linéaire equivariant par décalage peut être représenté par une convolution !



Rappel: Filtrage

► **Filtrage:**

- Forme une nouvelle image dont les pixels sont une combinaison des valeurs originales

► **BUT:**

- Extraire des informations utiles de l'image (lignes, coins, morceaux)
- Modifier ou améliorer les propriétés
- (suppression du bruit, super-résolution)

Convolution

Commutative: $f * g = g * f$

Associative: $(f * g) * h = f * (g * h)$

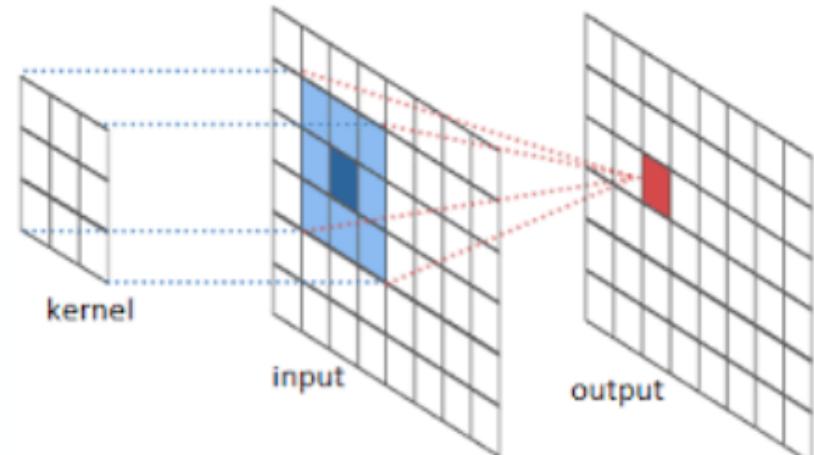
Distributive: $(f + g) * h = f * h + g * h$

Linear: $(af + bg) * h = af * h + bg * h$

Shift Invariant: $f(x+t) * h = (f * h)(x+t)$

Differentiation rule:

$$\frac{\partial}{\partial x}(f * g) = \frac{\partial f}{\partial x} * g$$



Convolution

Commutative: $f * g = g * f$

Associative: $(f * g) * h = f * (g * h)$

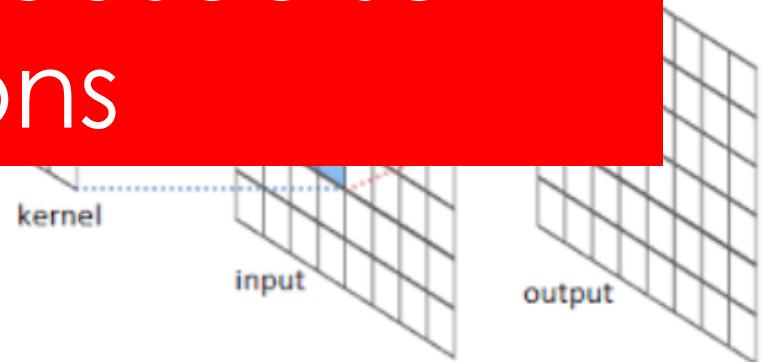
Distributive: $(f + g) * h = f * h + g * h$

Linear

Shift

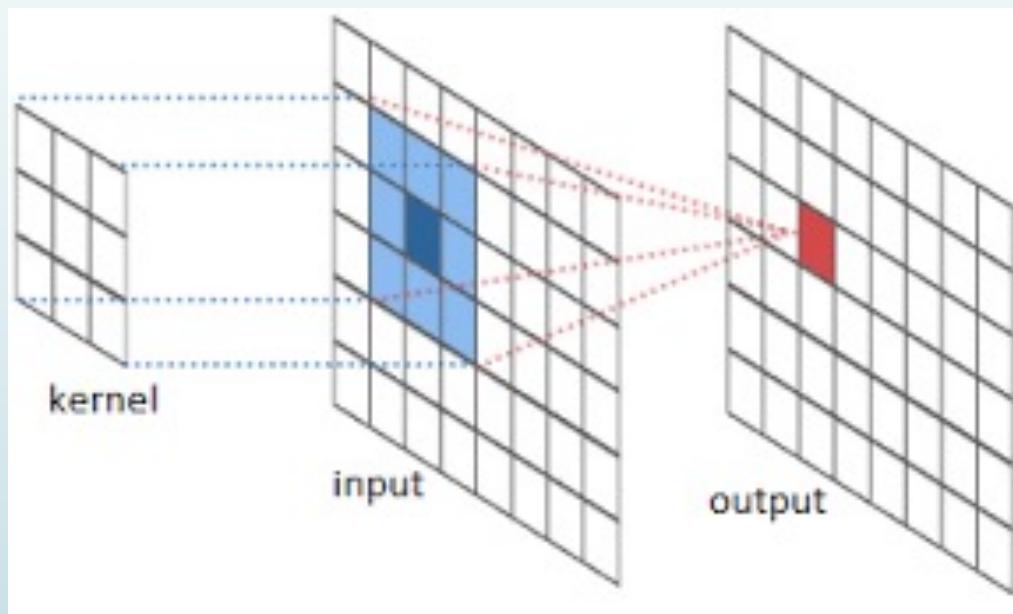
Differ

La plupart de la vision
moderne est basée sur
les convolutions



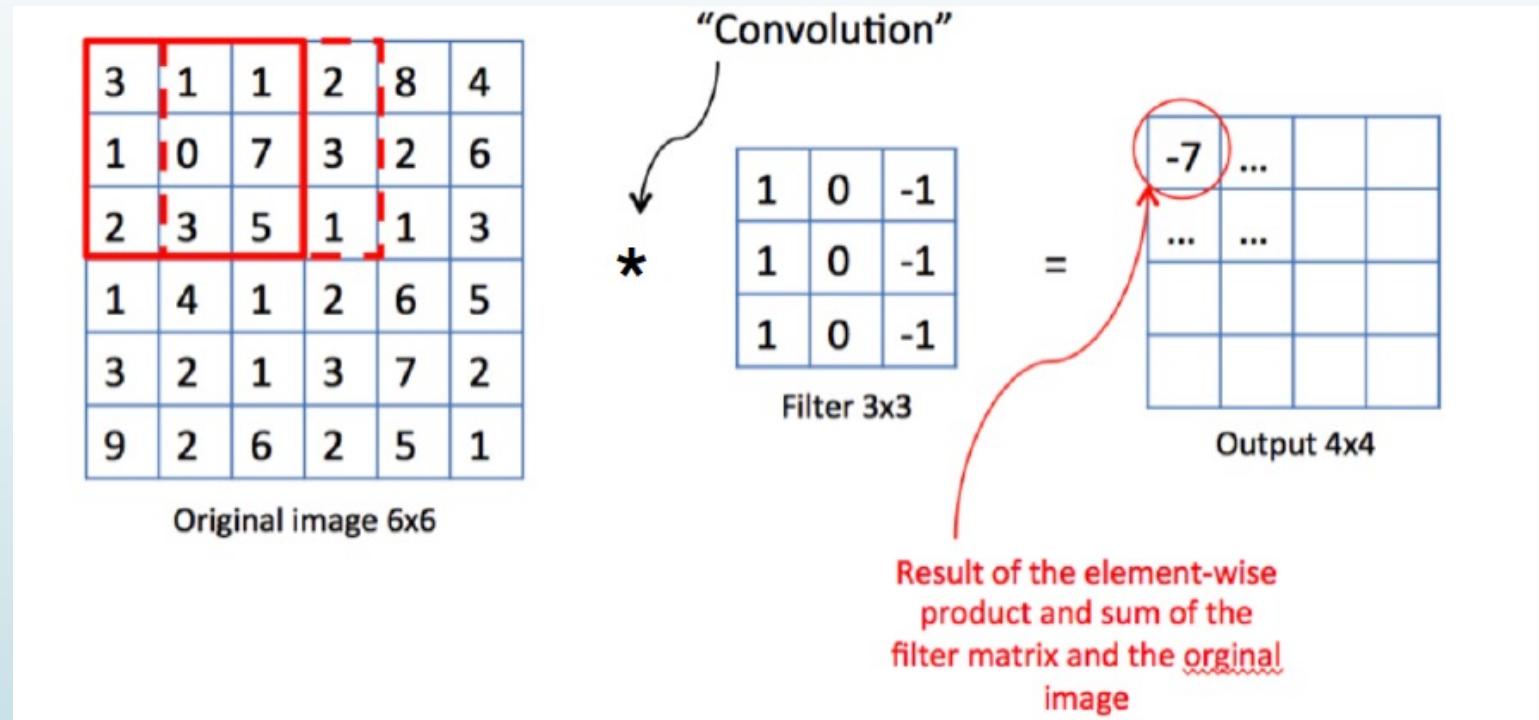
Convolution

- ▶ nombreux filtres standards peuvent être exprimées à l'aide de convolutions 2D, comme le lissage et la netteté des images et la détection des bords.
- ▶ La convolution en 2D utilise deux 'images', l'une est **image d'entrée** et l'autre, appelée **noyau**, sert de **filtre**.
- ▶ l'image de sortie est produite en faisant glisser le noyau sur l'image d'entrée.



Convolution

- La convolution est le processus d'ajout de chaque élément de l'image à ses voisins locaux, pondérés par le noyau.



Convolution pour signaux discrets 2D

Définition d'un filtre comme étant une convolution:

$$(f * g)(x, y) = \sum_{i,j=-\infty}^{\infty} f(i, j)I(x - i, y - j)$$

Image filtrée filtre entrée

Convolution pour signaux discrets 2D

Définition d'un filtre comme étant une convolution:

$$(f * g)(x, y) = \sum_{i,j=-\infty}^{\infty} f(i, j)I(x - i, y - j)$$

Image filtrée filtre entrée

Si le filtre $f(i, j)$ est non nul seulement pour $-1 \leq i, j \leq 1$, on a:

$$(f * g)(x, y) = \sum_{i,j=-1}^{1} f(i, j)I(x - i, y - j)$$

Le filtre boîte vu précédemment est la représentation matricielle de $f(i, j)$

Exemples de convolutions



Original

•0	•0	•0
•0	•1	•0
•0	•0	•0

=



Filtered
(no change)

Shifted right by one pixel:



Original

•0	•0	•0
•0	•0	•1
•0	•0	•0

=

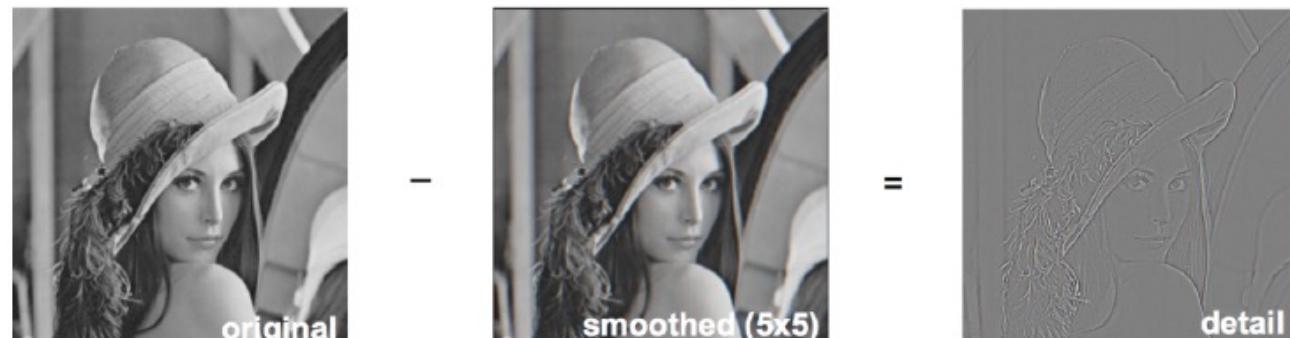


Shifted right
By 1 pixel

Exemples de convolution

- ▶ Un filtre de **netteté** peut être décomposé en deux étapes: il prend une image lissée, la soustrait de l'image originale pour obtenir les « **détails** » de l'image et ajoute les « **détails** » à l'image originale.

Step 1: Original - Smoothed = "Details"



$$\begin{array}{|c|c|c|} \hline •0 & •0 & •0 \\ \hline •0 & •1 & •0 \\ \hline •0 & •0 & •0 \\ \hline \end{array}$$

$$- \quad \frac{1}{9} \begin{array}{|c|c|c|} \hline •1 & •1 & •1 \\ \hline •1 & •1 & •1 \\ \hline •1 & •1 & •1 \\ \hline \end{array}$$

Exemples de convolution

- ▶ Un filtre de **netteté** peut être décomposé en deux étapes: il prend une image lissée, la soustrait de l'image originale pour obtenir les « **détails** » de l'image et ajoute les « **détails** » à l'image originale.

Step 2: Original + "Details" = Sharpened



$$\begin{array}{ccc} \begin{matrix} \bullet 0 & \bullet 0 & \bullet 0 \\ \bullet 0 & \bullet 1 & \bullet 0 \\ \bullet 0 & \bullet 0 & \bullet 0 \end{matrix} & + & \begin{matrix} \bullet 0 & \bullet 0 & \bullet 0 \\ \bullet 0 & \bullet 1 & \bullet 0 \\ \bullet 0 & \bullet 0 & \bullet 0 \end{matrix} - \frac{1}{9} \begin{matrix} \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \end{matrix} = \begin{matrix} \bullet 0 & \bullet 0 & \bullet 0 \\ \bullet 0 & \bullet 2 & \bullet 0 \\ \bullet 0 & \bullet 0 & \bullet 0 \end{matrix} - \frac{1}{9} \begin{matrix} \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \end{matrix} \end{array}$$

Exemples de convolution

- ▶ **Filtre de lissage gaussien:** est un filtre de moyenne pondérée qui donne plus de poids aux pixels centraux et moins de poids aux voisins

$$G_{\sigma} \equiv \frac{1}{2\pi\sigma^2} \exp\left\{-\frac{x^2+y^2}{2\sigma^2}\right\}$$



original

sigma = 3

sigma = 10

Exemples

Name	Kernel	Image Result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Mean Blur	$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$	
Laplacian	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
Gaussian Blur	$\begin{bmatrix} 1/16 & 2/16 & 1/16 \\ 2/16 & 4/16 & 2/16 \\ 1/16 & 2/16 & 1/16 \end{bmatrix}$	

Convolution

Commutative: $f * g = g * f$

Associative: $(f * g) * h = f * (g * h)$

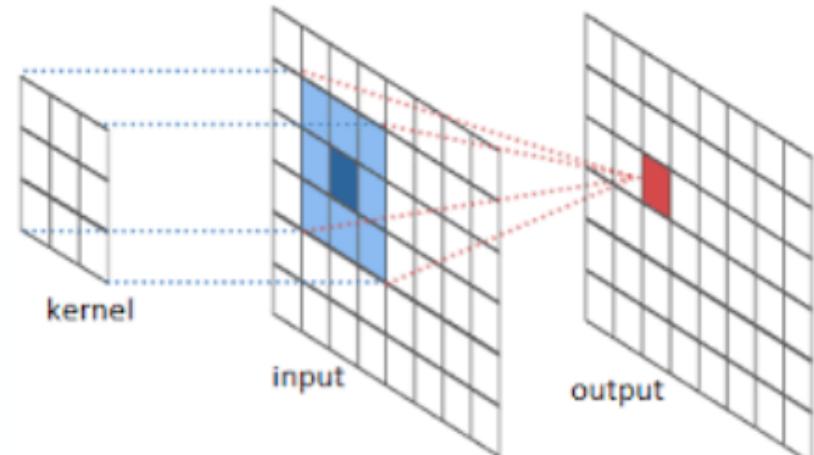
Distributive: $(f + g) * h = f * h + g * h$

Linear: $(af + bg) * h = af * h + bg * h$

Shift Invariant: $f(x+t) * h = (f * h)(x+t)$

Differentiation rule:

$$\frac{\partial}{\partial x}(f * g) = \frac{\partial f}{\partial x} * g$$



Separable filters

Un filtre 2D est dit ‘séparable’ si il peut s’écrire comme le produit d’une matrice colonne et d’une matrice ligne

example:
box filter

$$\begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} = \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline \end{array}$$

column

row

Une convolution 2D avec un filtre séparable est équivalente à deux convolution 1D

Une image à $N \times N$ pixels, un filtre à $M \times M$ éléments

Quel est le coût de calcul d'une convolution? (**avec un filtre séparable**)

$$2 \times M^2 \times N$$

Exemples de filtres séparables

- ▶ **Box-filter:** utilisé comme filtre de lissage.

Simple box filter

$$\frac{1}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} * \frac{1}{3} [1 \quad 1 \quad 1] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

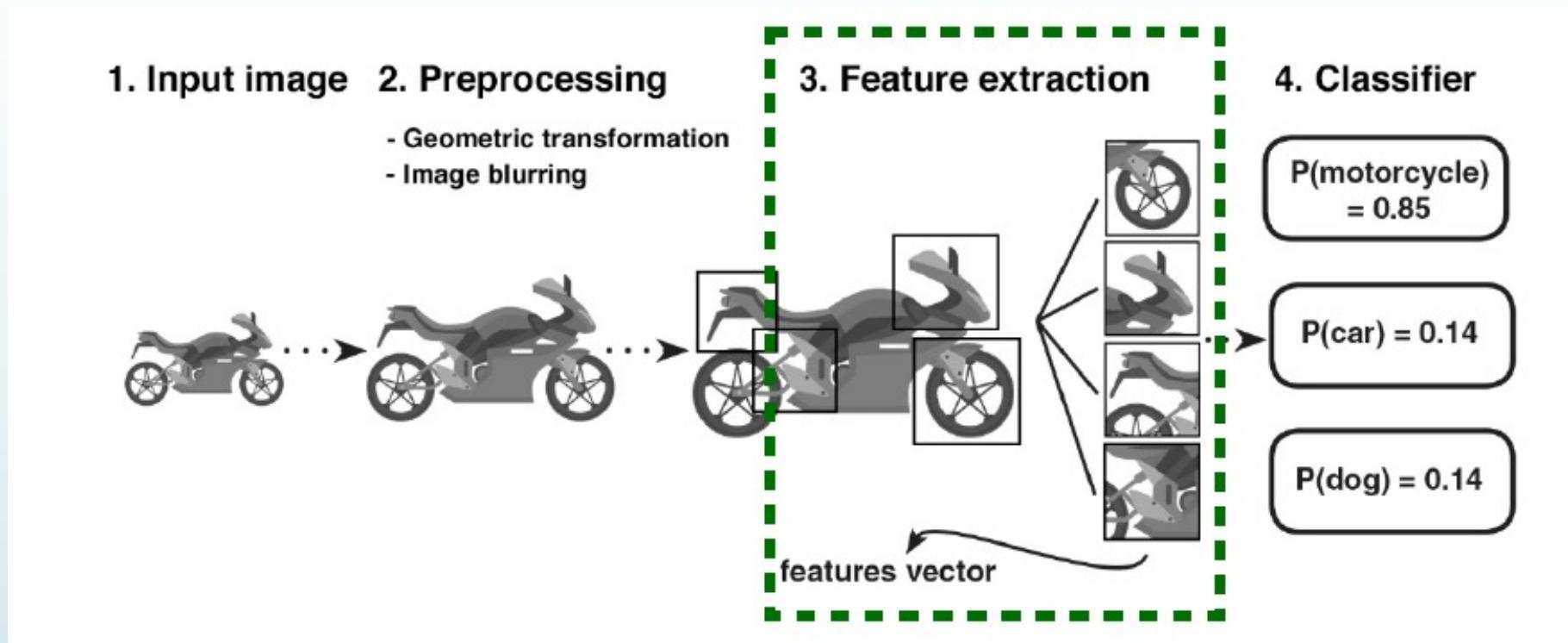
- ▶ **Opérateur Sobel** (couramment utilisé pour la détection des bords.)

Simple Gaussian

$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * [1 \quad 0 \quad -1] = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

Finite diff operator

Pipeline

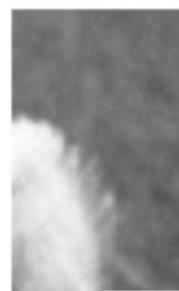


Détection des bords

- ▶ **Les bords** sont les points d'une image où **la luminosité de l'image change brusquement ou présente des discontinuités**. Ces discontinuités correspondent généralement à :
 - ▶ Discontinuités en profondeur
 - ▶ Discontinuités dans l'orientation de la surface
 - ▶ Modifications des propriétés des matériaux
 - ▶ Variations dans l'éclairage de la scène
- ▶ Les bords sont **importants** pour deux raisons principales.
 - ▶ La plupart des **informations sémantiques et de forme peuvent en être déduites**, ce qui nous permet d'effectuer la reconnaissance d'objets et d'analyser les perspectives et la géométrie d'une image.
 - ▶ Ils sont une **représentation plus compacte que les pixels**.

Détection des bords

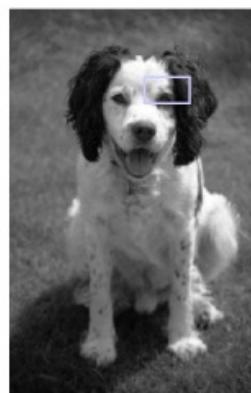
Boundaries of objects



Boundaries of Lighting

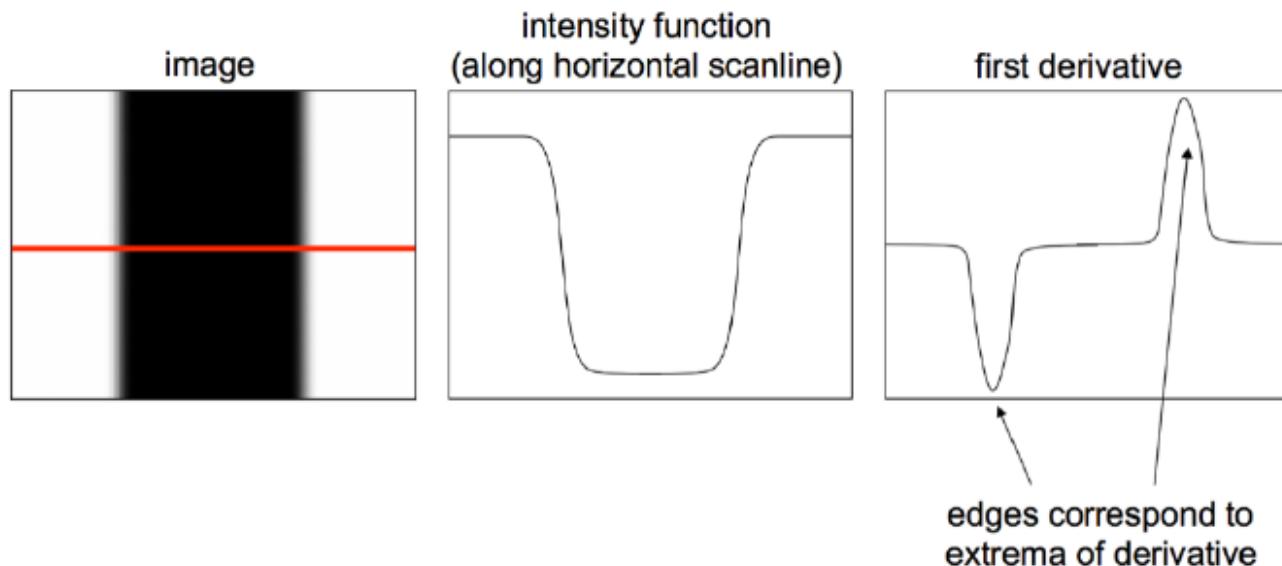


Boundaries of Material Properties



Caractérisation des arêtes

- ▶ **localiser les bords** à partir du profil d'intensité d'une image le long d'une ligne ou d'une colonne de l'image. Partout où il y a un **changement rapide dans la fonction d'intensité indique une arête**, comme on le voit où la première dérivée de la fonction a un extrema.



Dérivées partielles avec convolution

Pour une fonction 2D $f(x, y)$ la dérivée partielle est:

$$\frac{\partial f(x, y)}{\partial x} = \lim_{\varepsilon \rightarrow 0} \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon}$$

Pour des éléments discrets on approxime avec des différences finies:

$$\frac{\partial f(x, y)}{\partial x} \approx \frac{f(x+1, y) - f(x, y)}{1}$$

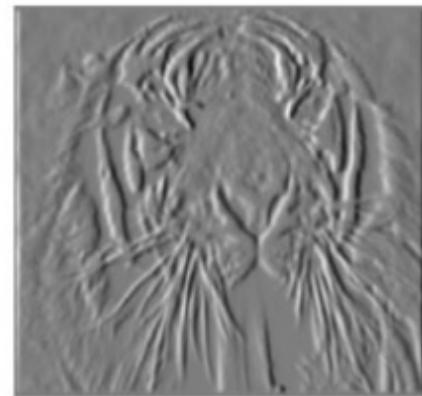
En terme d'implémentation, quel est le filtre associé au calcul d'une telle différence?

Source: K. Grauman

Dérivées partielles d'une image

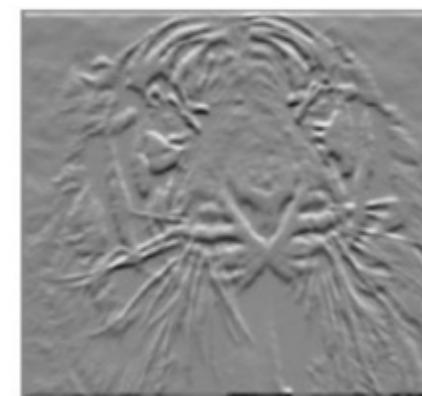
-1	0	1
----	---	---

$$\frac{\partial f(x, y)}{\partial x}$$



1
0
-1

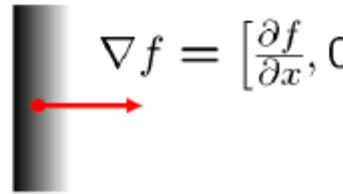
$$\frac{\partial f(x, y)}{\partial y}$$

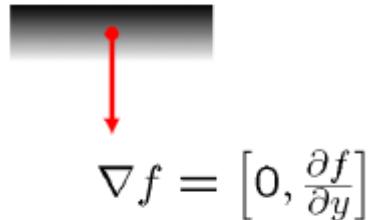


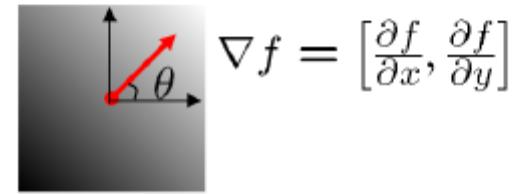
Laquelle correspond à des changements par rapport à x?

Gradient d'une image

· Le gradient d'une image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$


$$\nabla f = \left[\frac{\partial f}{\partial x}, 0 \right]$$


$$\nabla f = \left[0, \frac{\partial f}{\partial y} \right]$$


$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

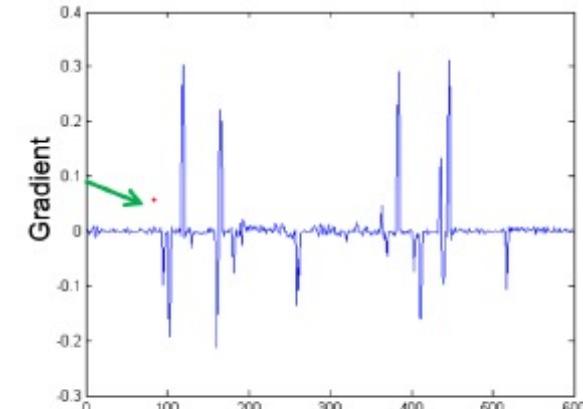
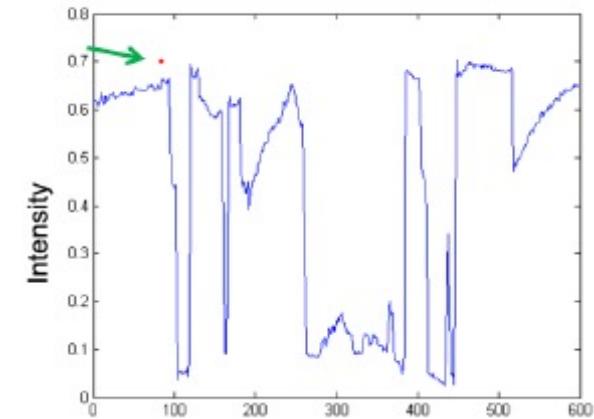
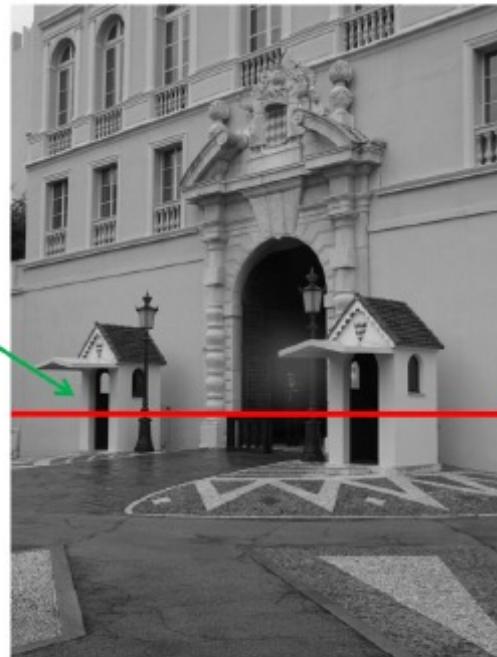
Le gradient est la direction vers laquelle l'intensité augmente le plus

L'intensité d'une arrête est donnée par la norme du gradient

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Source: Steve Seitz

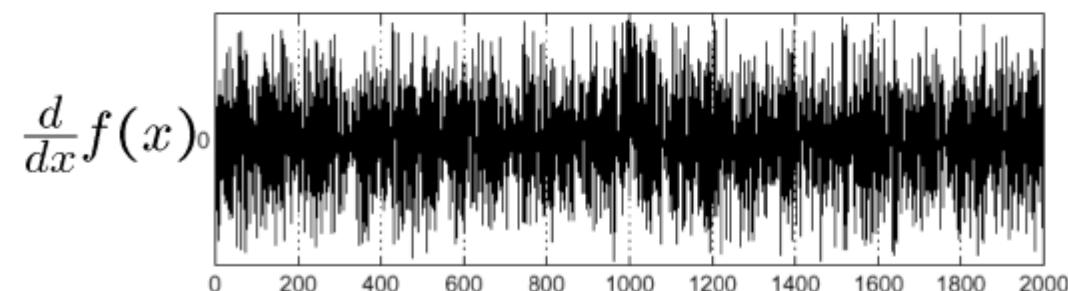
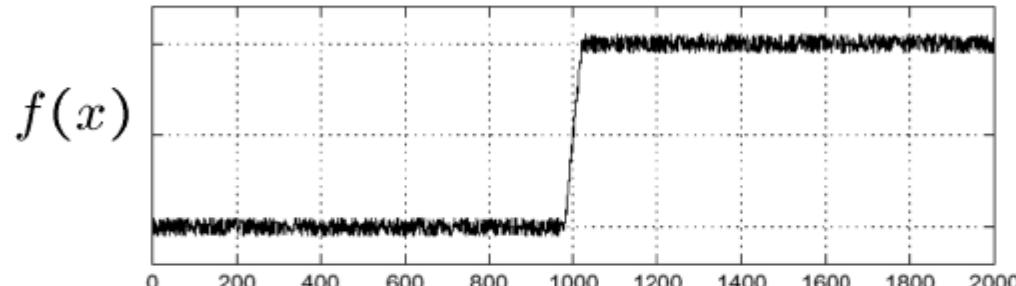
Profil d'intensité



Source: D. Hoiem

Effet du bruit

Considérons une ligne de l'image. On représente l'intensité du signal sur cette ligne:



D'après le gradient, où est l'arrêté?

nal

ce: S. Seitz

Effect of noise

Les différences finies sont **très sensibles au bruit**:

- Le bruit rend certains pixels très différents de leurs voisins.
- Plus le bruit est grand plus c'est un problème.

Solution: ????

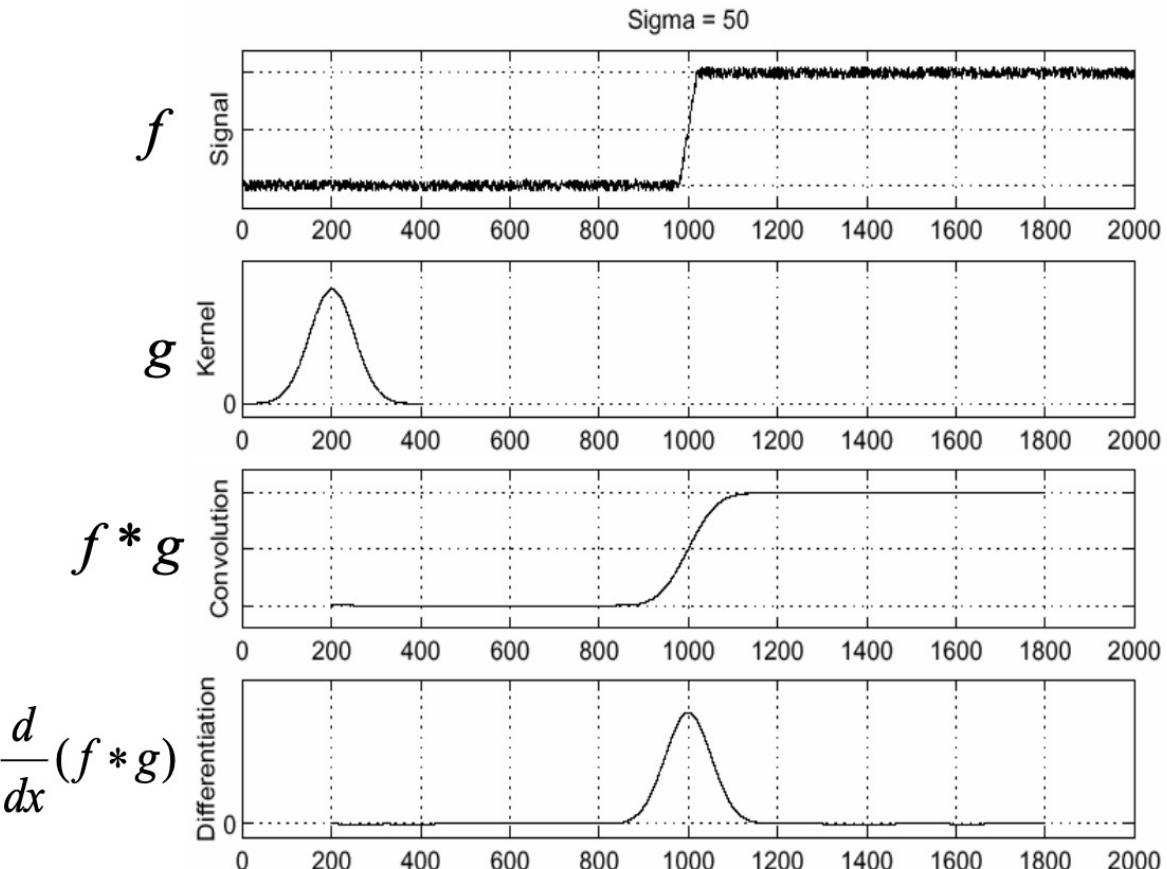
Effect of noise

Les différences finies sont **très sensibles au bruit**:

- Le bruit rend certains pixels très différents de leurs voisins.
- Plus le bruit est grand plus c'est un problème.

Solution: **lissage** (on force les pixels à ne pas trop dévier de la valeur de leurs voisins)

Solution : Lissage



Pour trouver des arrêtes: regarder les valeurs de $|\partial_x(f * g)|$

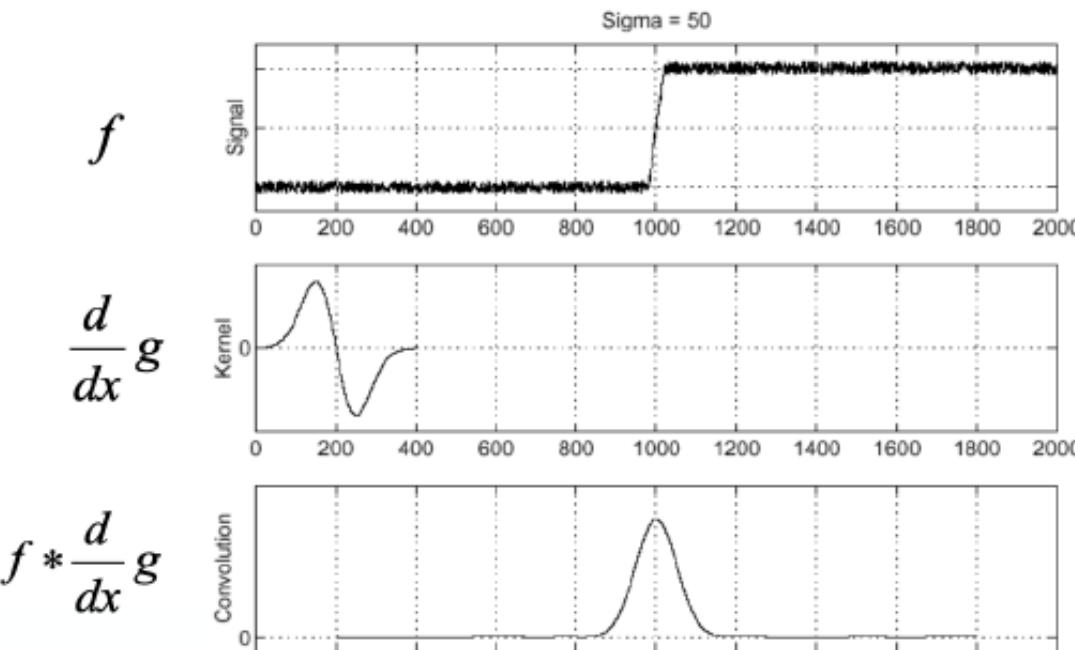
Source: S. Seitz

Edge detection via Convolution

- Associativité:

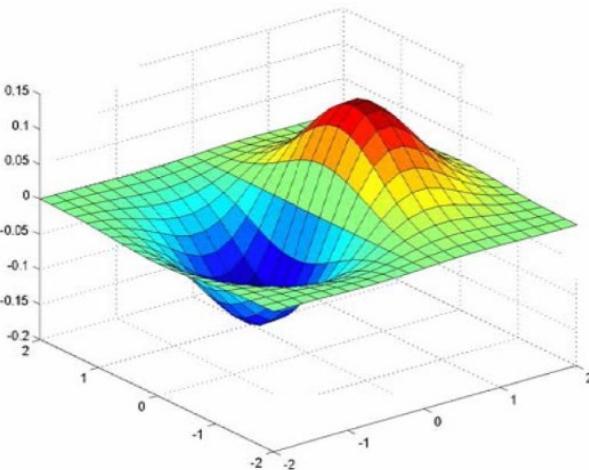
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx}g$$

- On évite un calcul!

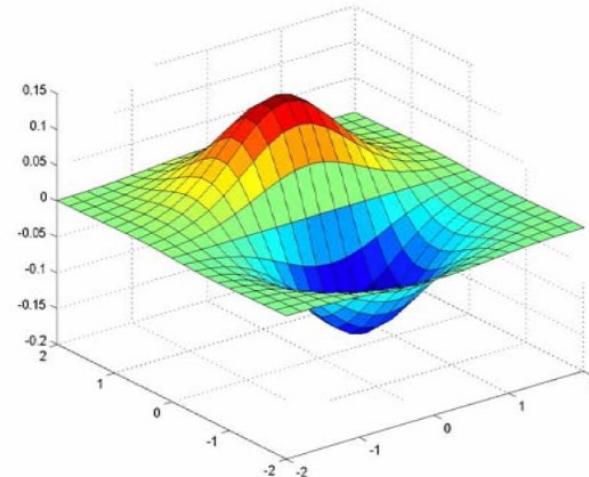


Source: S. Seitz

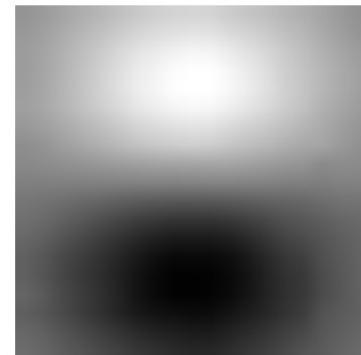
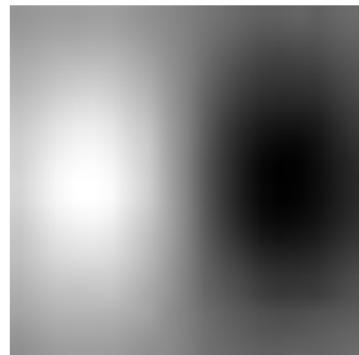
Dérivé du filtre gaussien



x-direction



y-direction



Laquelle trouve les arrêtes horizontales???

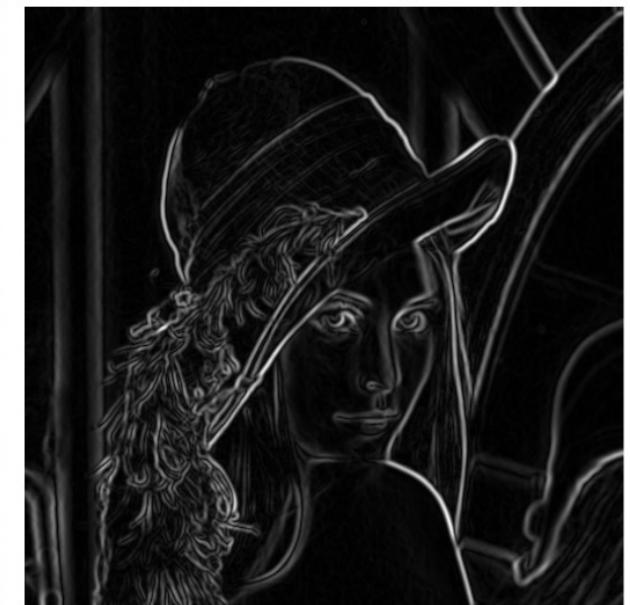
Dérivées du filtre gaussien



X-Derivative of Gaussian



Y-Derivative of Gaussian



Gradient Magnitude

DéTECTEURS de bord

Opérateur de Prewitt: filtre boîte convolé avec un opérateur de dérivation

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} = \boxed{\text{Finite diff operator}} * \boxed{\text{Simple box filter}}$$
$$= \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Opérateur de Sobel: filtre Gaussien convolé avec un opérateur de dérivation

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \boxed{\text{Finite diff operator}} * \boxed{\text{Simple Gaussian}}$$
$$= \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

Detection d'arrêté

```
1 #import the required libraries
2 import numpy as np
3 import cv2
4 import matplotlib.pyplot as plt
5 %matplotlib inline
6 #read the image
7 image = cv2.imread('coins.jpg')
8 #calculate the edges using Canny edge algorithm
9 edges = cv2.Canny(image,100,200)
10 #plot the edges
11 plt.imshow(edges)
```

Edge Detection.py hosted with ❤ by GitHub

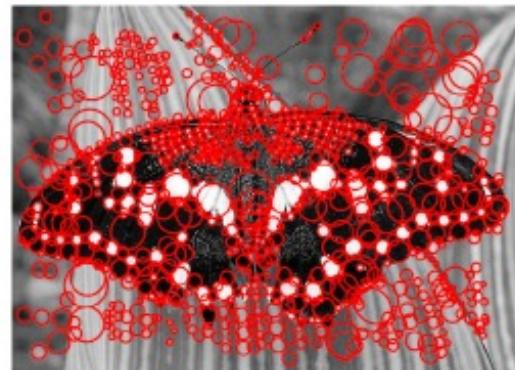
[view raw](#)



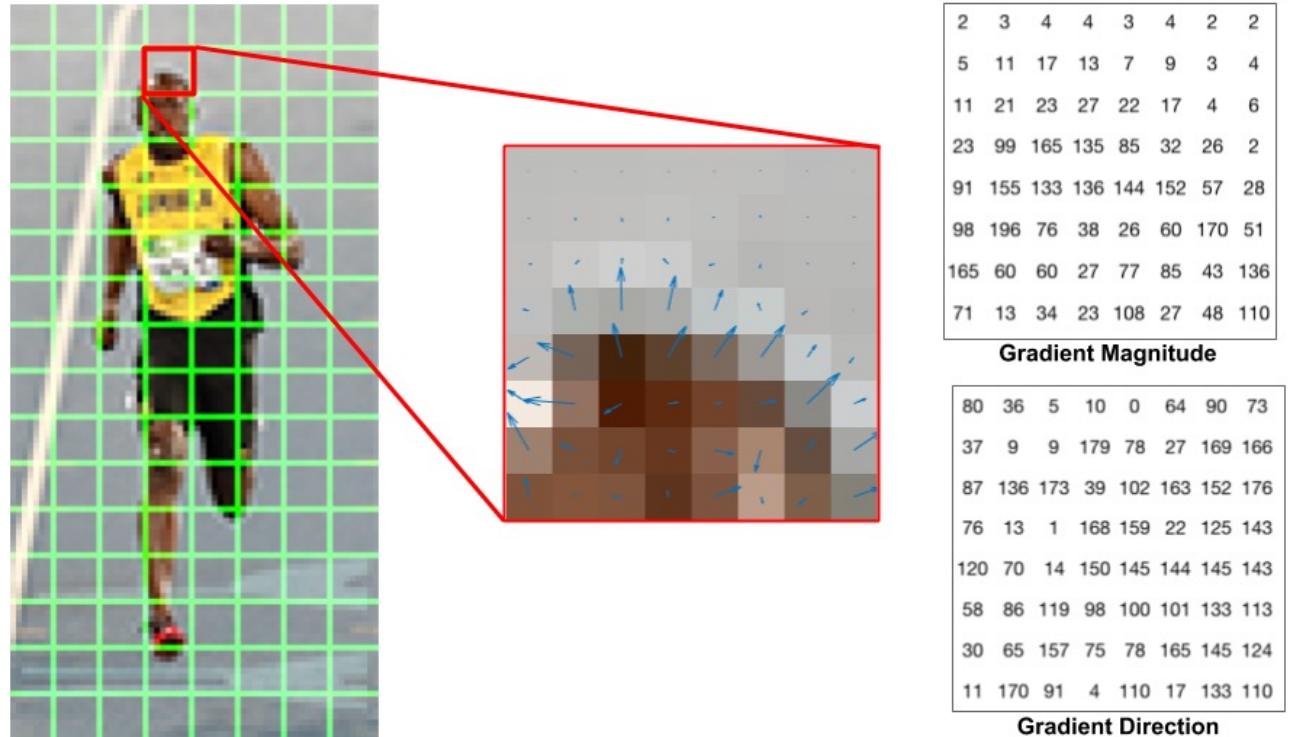
J. Canny, [*A Computational Approach To Edge Detection*](#), IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

DéTECTEURS DE COINS/blob

- ▶ Les bords sont utiles en tant qu'entités locales, mais les coins et les petites zones (blobs) sont généralement plus utiles dans les tâches de vision par ordinateur. Les détecteurs d'objets blob peuvent être construits en étendant l'idée de base du détecteur de bord dont nous venons de parler.



Exemple : histogramme de gradients

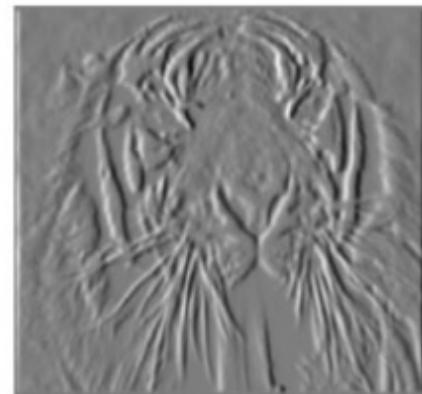


Source: <https://learnopencv.com/histogram-of-oriented-gradients/>

Dérivées partielles d'une image

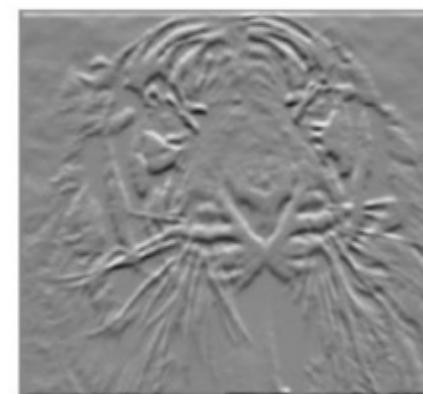
-1	0	1
----	---	---

$$\frac{\partial f(x, y)}{\partial x}$$



1
0
-1

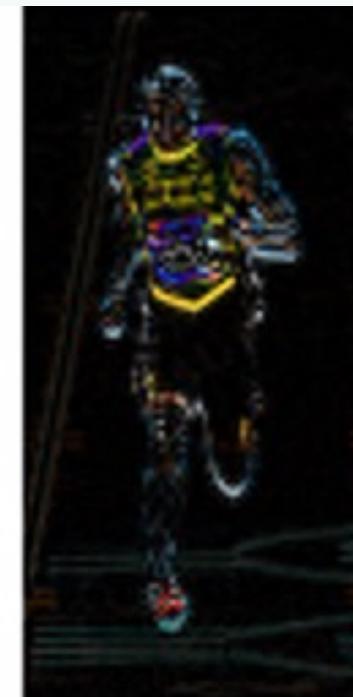
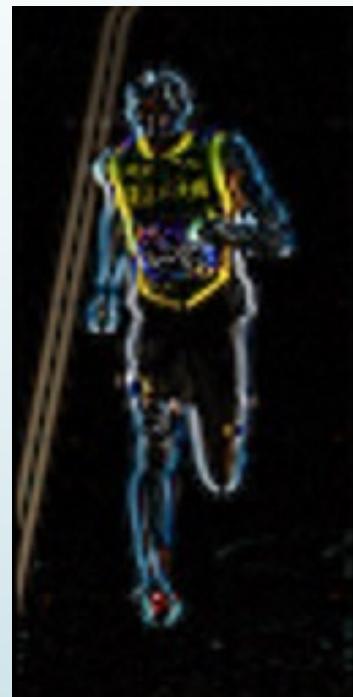
$$\frac{\partial f(x, y)}{\partial y}$$



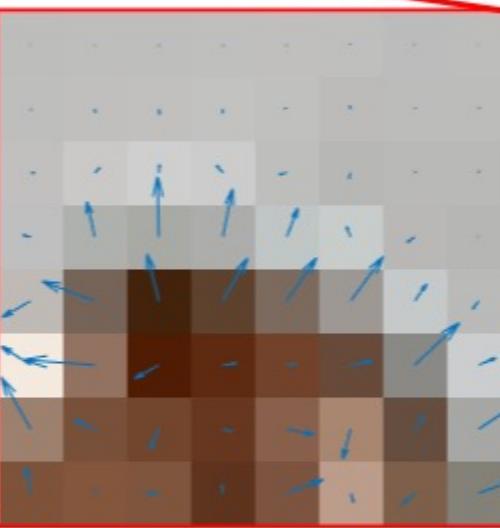
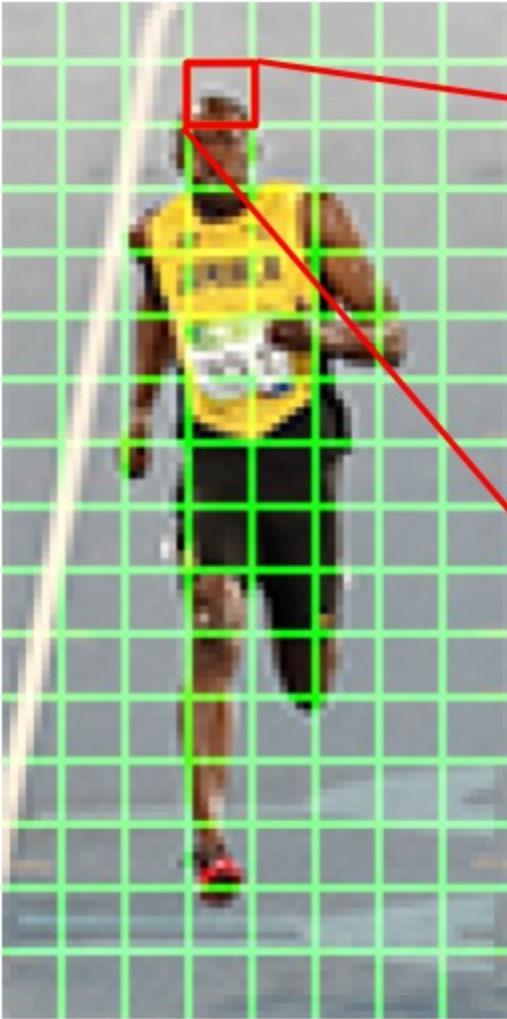
Which shows changes with respect to x?

Le gradient est une caractéristique significative

- À gauche : Valeur absolue du gradient x
- Centre : valeur absolue du gradient y
- À droite : Amplitude du gradient
- Supprime toutes les parties constantes.
- Grande magnitude : **arrête**
- Direction du gradient : **Direction de l'arrêté**



Regarder les patches 8x8



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

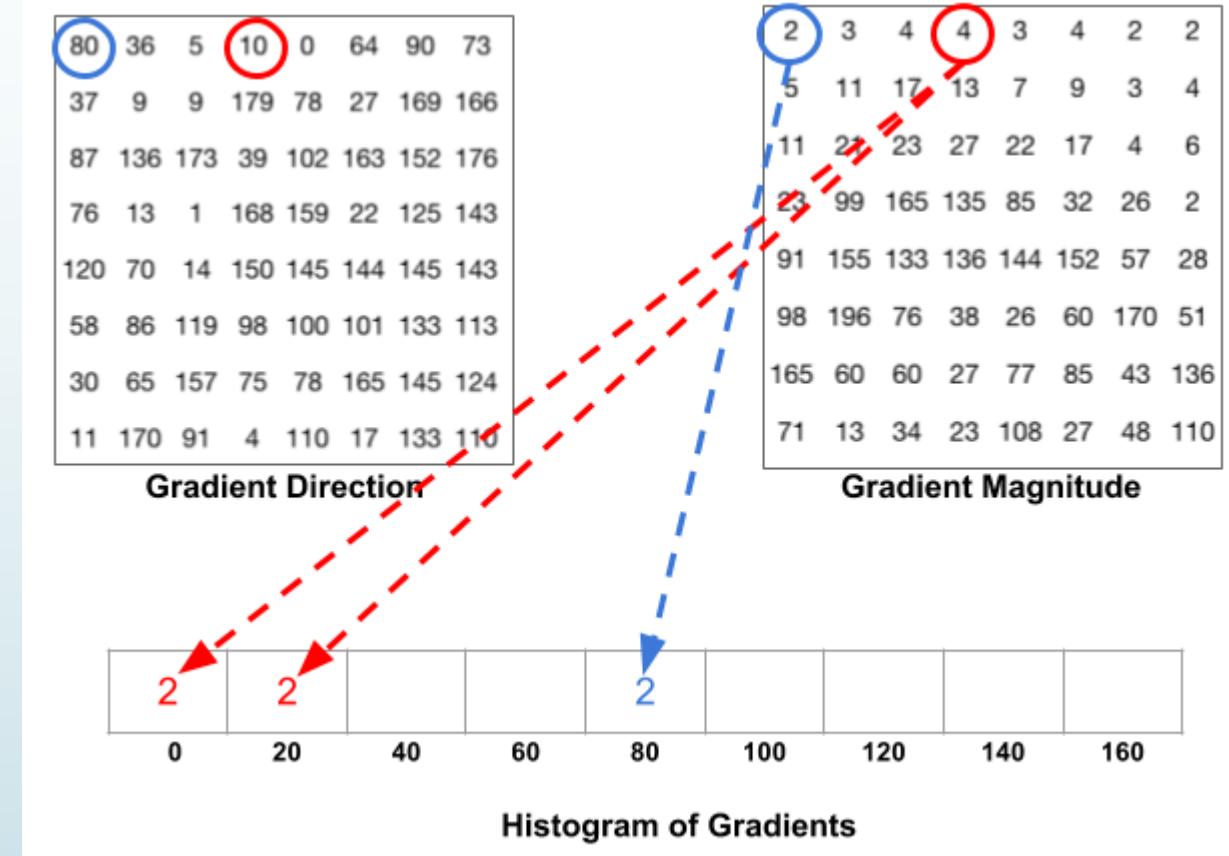
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

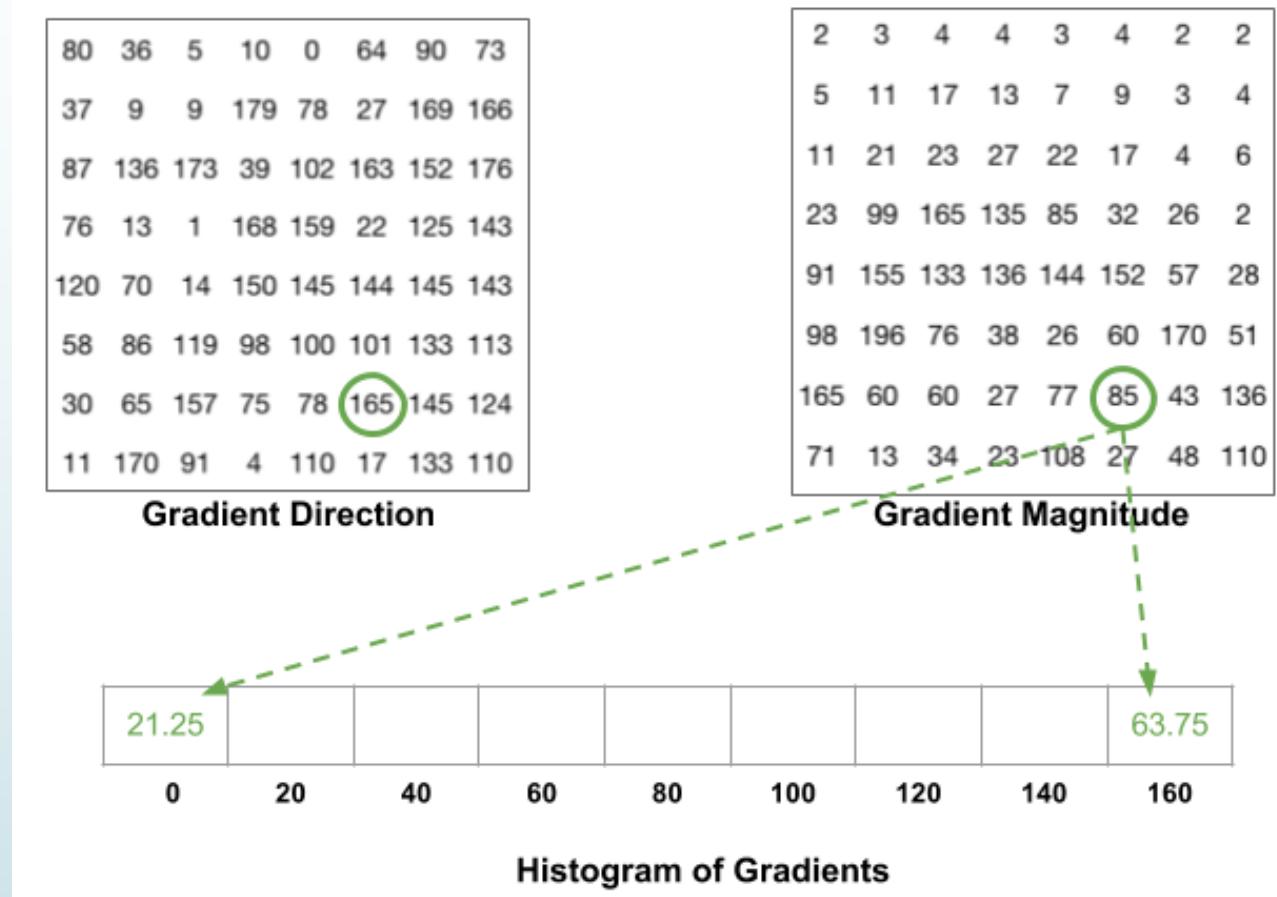
Comportement moyen de chaque patch: Histogramme

- ▶ Distribution de la direction sur le patch.
- ▶ Fonction locale
- ▶ Plus robuste que toute la direction concaténée.
- ▶ Peut détecter les bords et les coins.



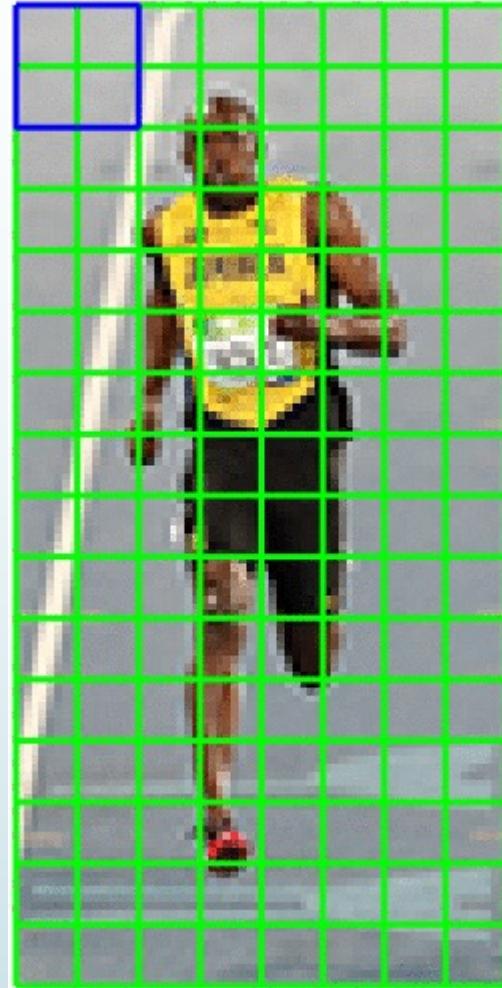
Une meilleure façon de faire la moyenne

- ▶ Distribution de la direction sur le patch.
- ▶ Fonction locale
- ▶ Plus robuste que toute la direction concaténée.
- ▶ Peut détecter les bords et les coins.
- ▶ Moyenne par rapport à l'ampleur (contribution plus importante si grande magnitude)
- ▶ Donne un vecteur de taille 8



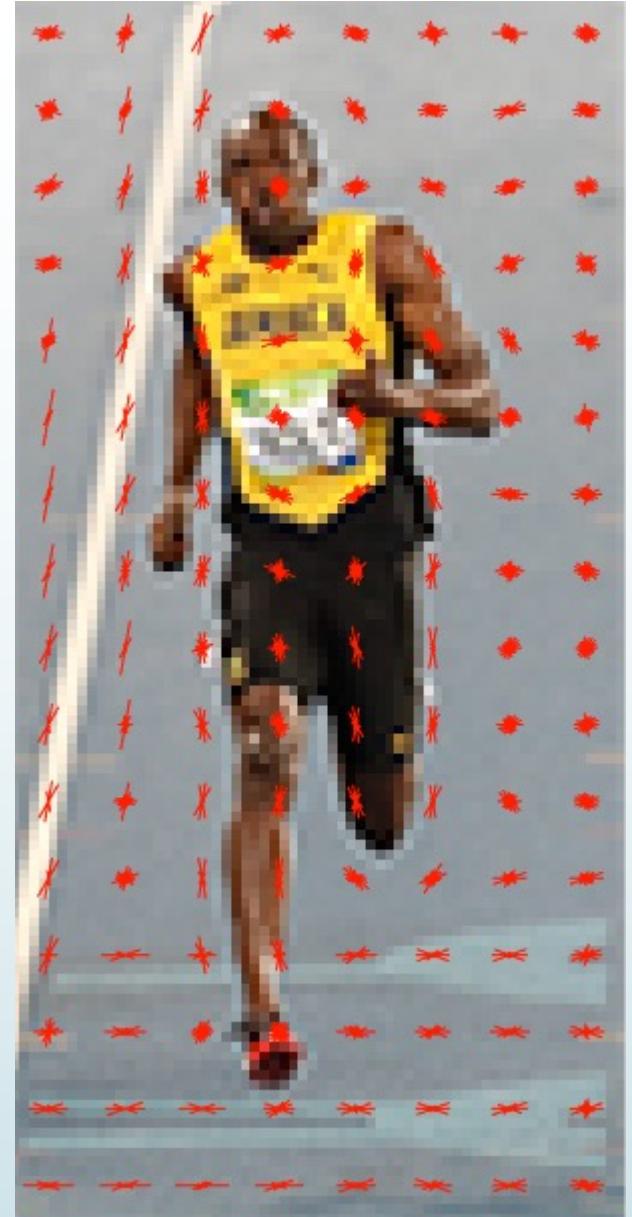
Normaliser les histogrammes

- ▶ À l'heure actuelle, les histogrammes peuvent avoir différentes échelles (nous venons d'additionner les magnitudes)
- ▶ Nous pourrions normaliser les valeurs
- ▶ Mieux vaut normaliser sur un patch plus grand!
- ▶ 16x16 patches donne 4 histogrammes.
- ▶ Normalisation L2 de ce vecteur de taille 36
- ▶ À la fin : 105 (patchs 16x16)
- ▶ Représentation finale : $36 \times 105 = 3780$ caractéristiques.



Que faire avec ça?

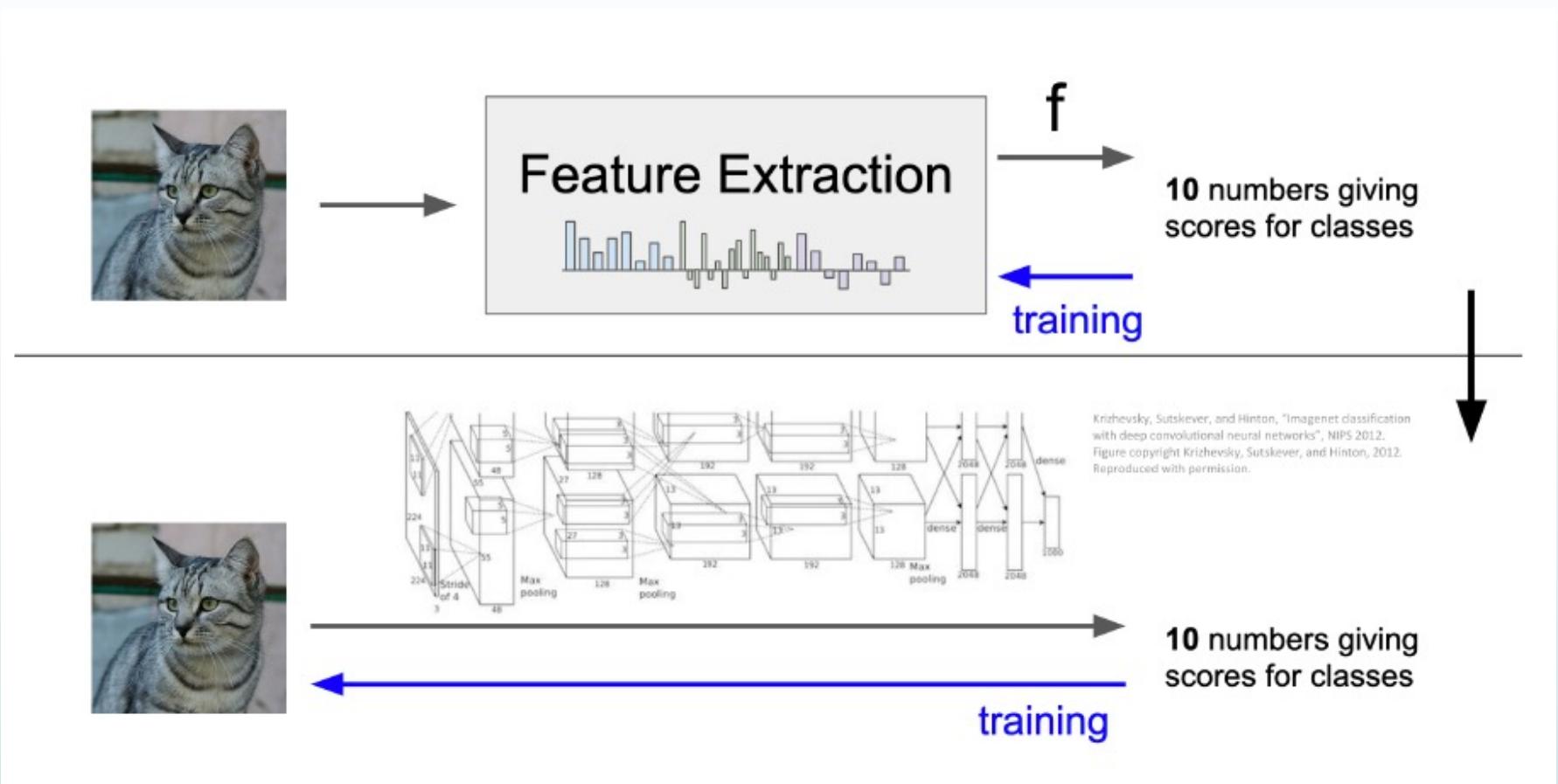
- ▶ Un classificateur linéaire (Log reg ou SVM) en plus de cela donnerait de bons résultats.
- ▶ Calculer des distances entre images (distance L2 entre les caractéristiques extraites)



Autres transformations de caractéristiques



Fonctionnalités d'image vs ConvNets



Apprentissage profond

- Dernière couche : classificateur linéaire
- couches cachées : apprentissage des fonctionnalités

For each sample

$$\{x_i, y_i\}$$

1. Predict
 - a. Forward pass

$$\hat{y} = wx_i$$

- b. Compute Loss

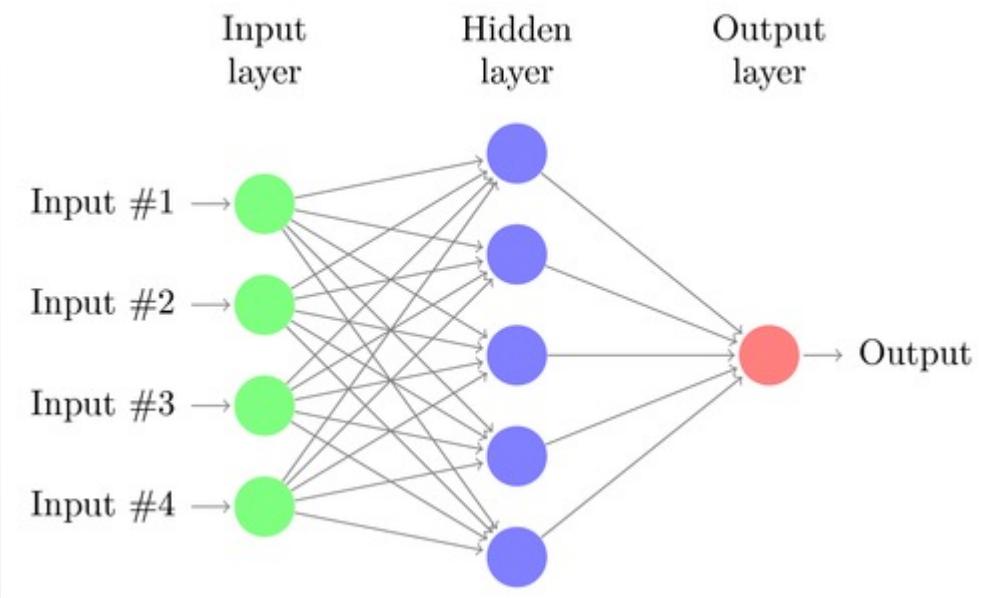
$$\mathcal{L}_i = \frac{1}{2}(y_i - \hat{y})^2$$

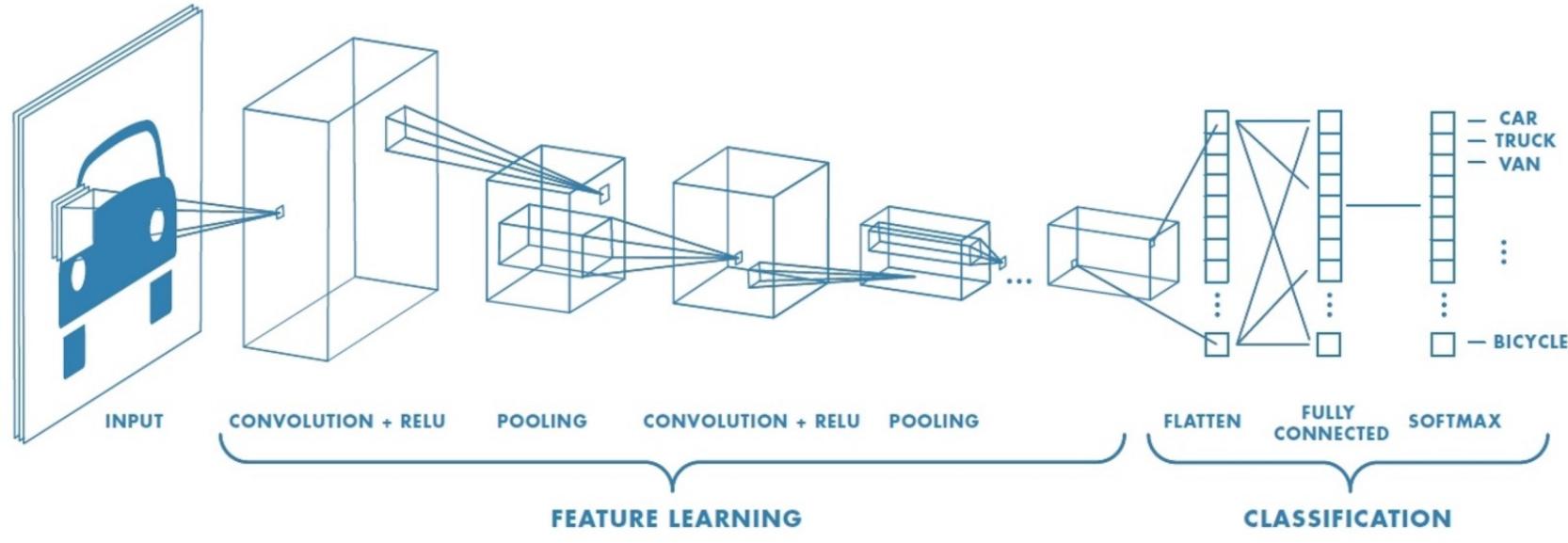
2. Update
 - a. Back Propagation

- b. Gradient update

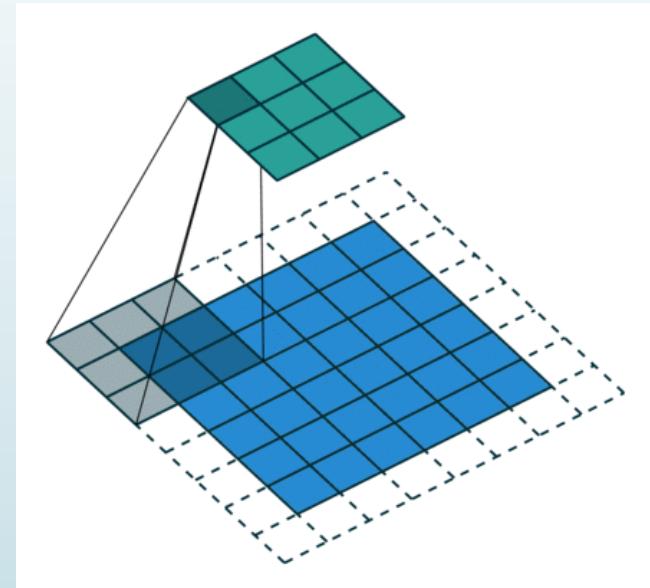
$$\frac{d\mathcal{L}_i}{dw} = -(y_i - \hat{y})x_i = \nabla w$$

$$w = w - \nabla w$$

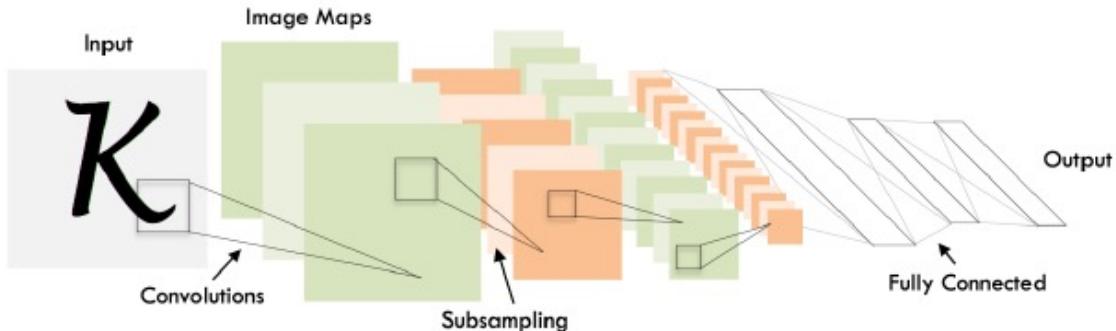




- ▶ Les poids des matrices de convolution sont appris!!!!
- ▶ Padding : ajoutez un peu de 0 (ou autre chose) autour de l'image.
- ▶ Si vous voulez en savoir plus (cours d'apprentissage de la représentation)



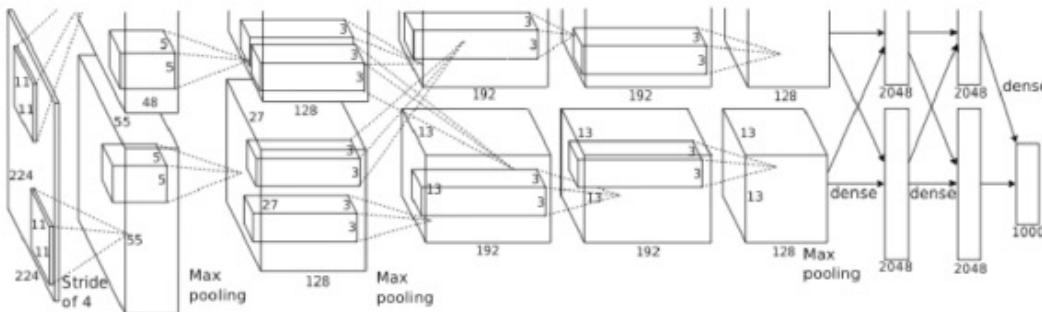
1998
LeCun et al.



of transistors
 10^6
pentium® II

of pixels used in training
 10^7

2012
Krizhevsky et al.



of transistors



GPUs

of pixels used in training

10^{14}

Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

ImageNet

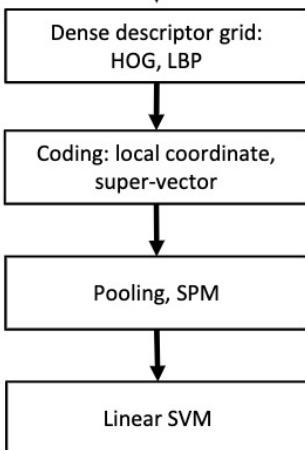
- ▶ ImageNet est une base de données d'images organisée selon la hiérarchie WordNet, dans laquelle chaque nœud de la hiérarchie est représenté par des centaines et des milliers d'images. Actuellement, nous avons en moyenne plus de cinq cents images par nœud



IMAGENET Large Scale Visual Recognition Challenge

Year 2010

NEC-UIUC

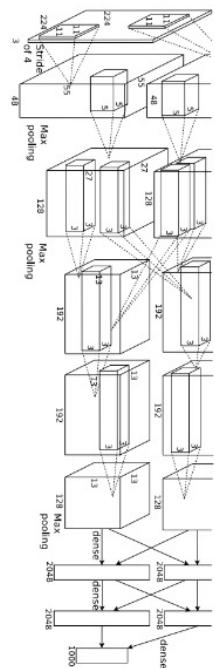


[Lin CVPR 2011]

Lion image by Swissfrog is licensed under CC BY 3.0

Year 2012

SuperVision



[Krizhevsky NIPS 2012]

Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Year 2014

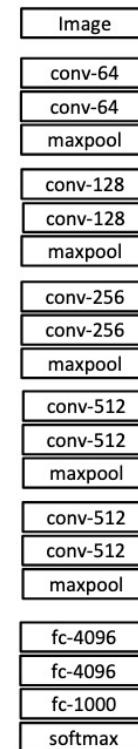
GoogLeNet

- Pooling
- Convolution
- Softmax
- Other



[Szegedy arxiv 2014]

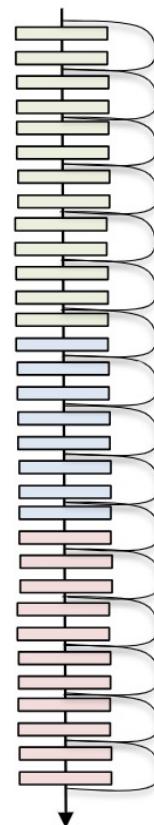
VGG



[Simonyan arxiv 2014]

Year 2015

MSRA



[He ICCV 2015]

AlexNet

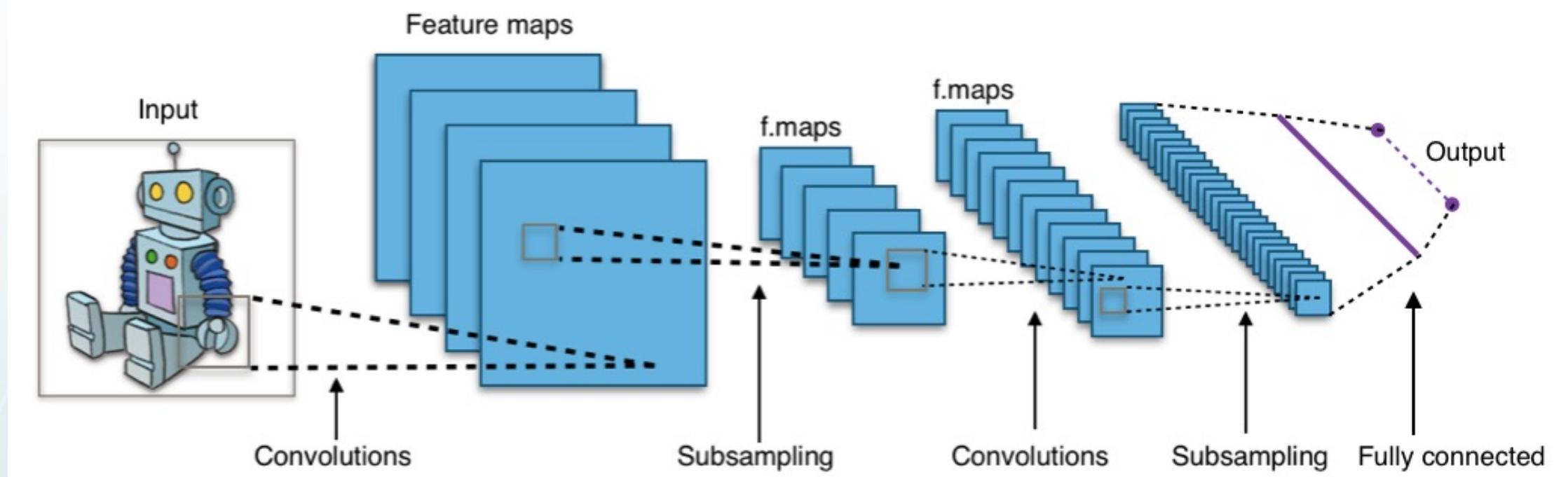
AlexNet est considéré comme l'un des papier les plus influent en vision



Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The

“AlexNet is considered one of the most influential papers published in computer vision, having spurred many more papers published employing CNNs and GPUs to accelerate deep learning.”

Réseaux de neurones convolutionels



Attention

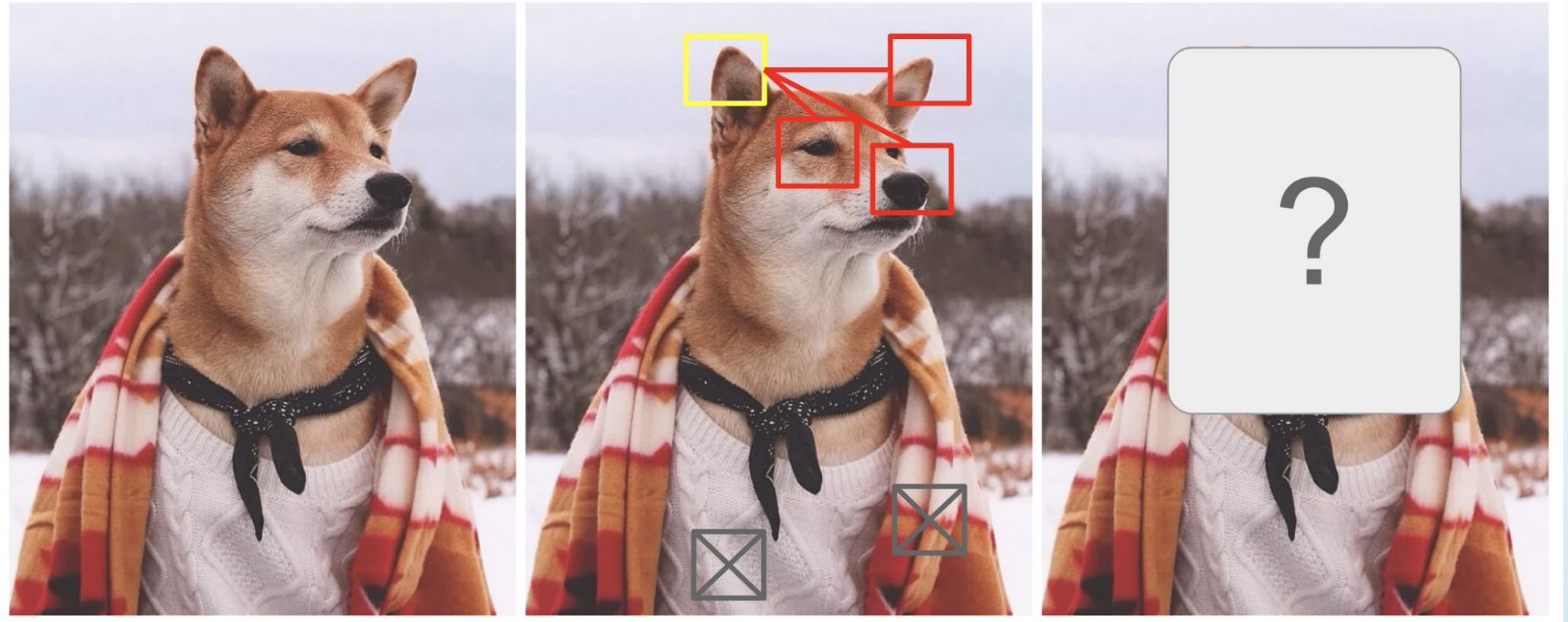
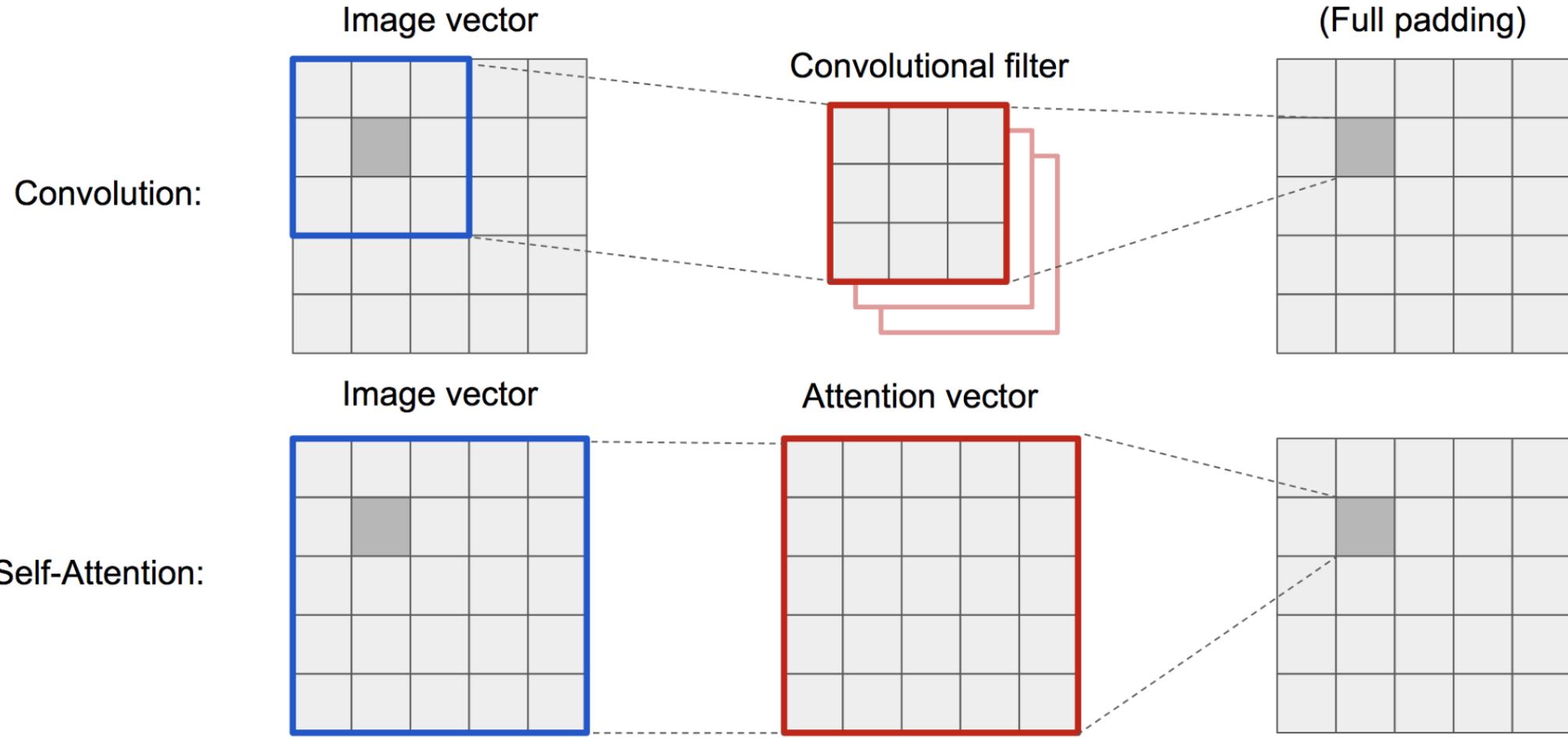


Fig. 1. Un Shiba Inu en tenue masculine. Le crédit de la photo originale va à Instagram @mensweardog.

Attention

- ▶ Utilisé à l'origine en NLP pour traiter les dépendances à long terme.
- ▶ Maintenant utilisé dans la vision aussi!
- ▶ Beaucoup de variations différentes autour d'une idée
- ▶ Repondération contextuelle des fonctionnalités.

(Auto-attention) Self Attention



Auto-Attention

Les caractéristiques x sont ramifiés en trois copies, correspondant aux concepts de clé, de valeur et de requête :

- ▶ Clé: $f(x) = W_f x$
- ▶ Requête: $g(x) = W_g x$
- ▶ Valeur: $h(x) = W_h x$

Ensuite, nous appliquons attention (softmax) par produit scalaire pour obtenir les caractéristiques induites par l'auto-attention :

$$\alpha_{i,j} = \text{softmax}\left(f(x_i), g(x_j)\right), \quad o_j = W_v \sum_i \alpha_{i,j} h(x_i)$$

Auto-Attention

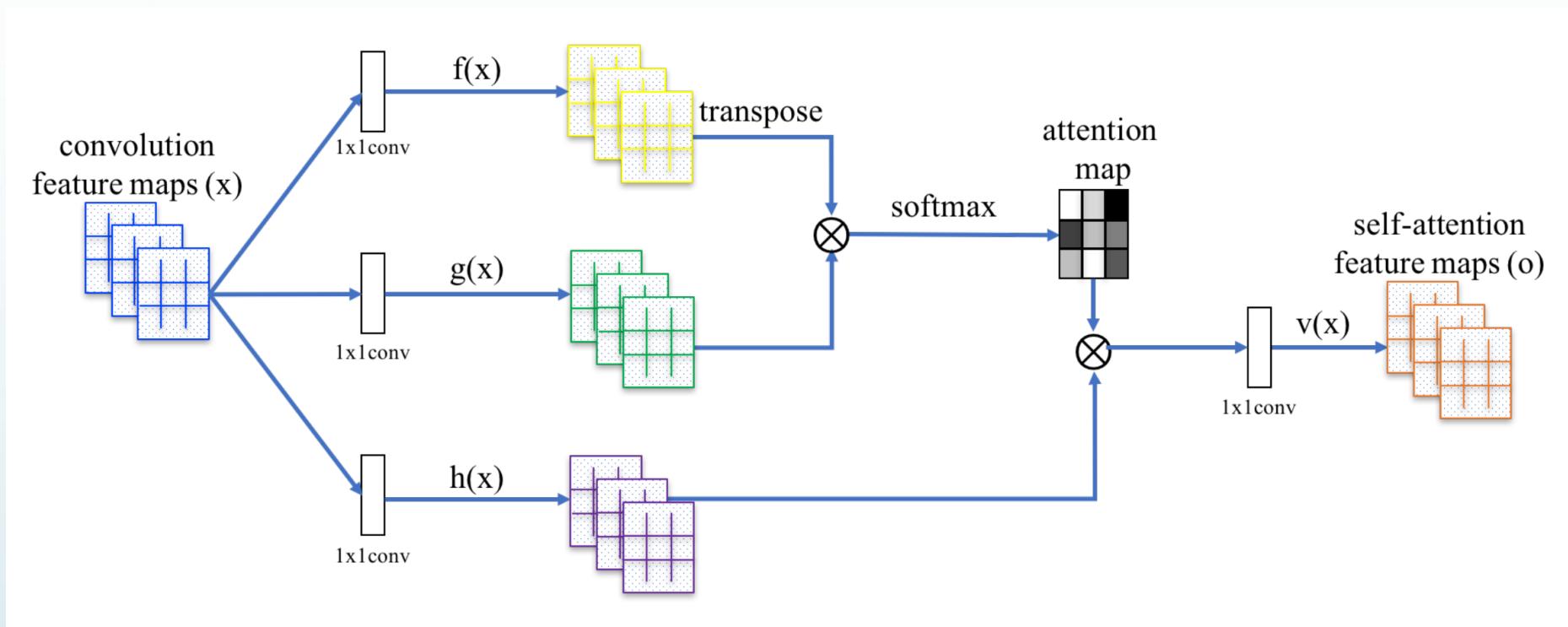
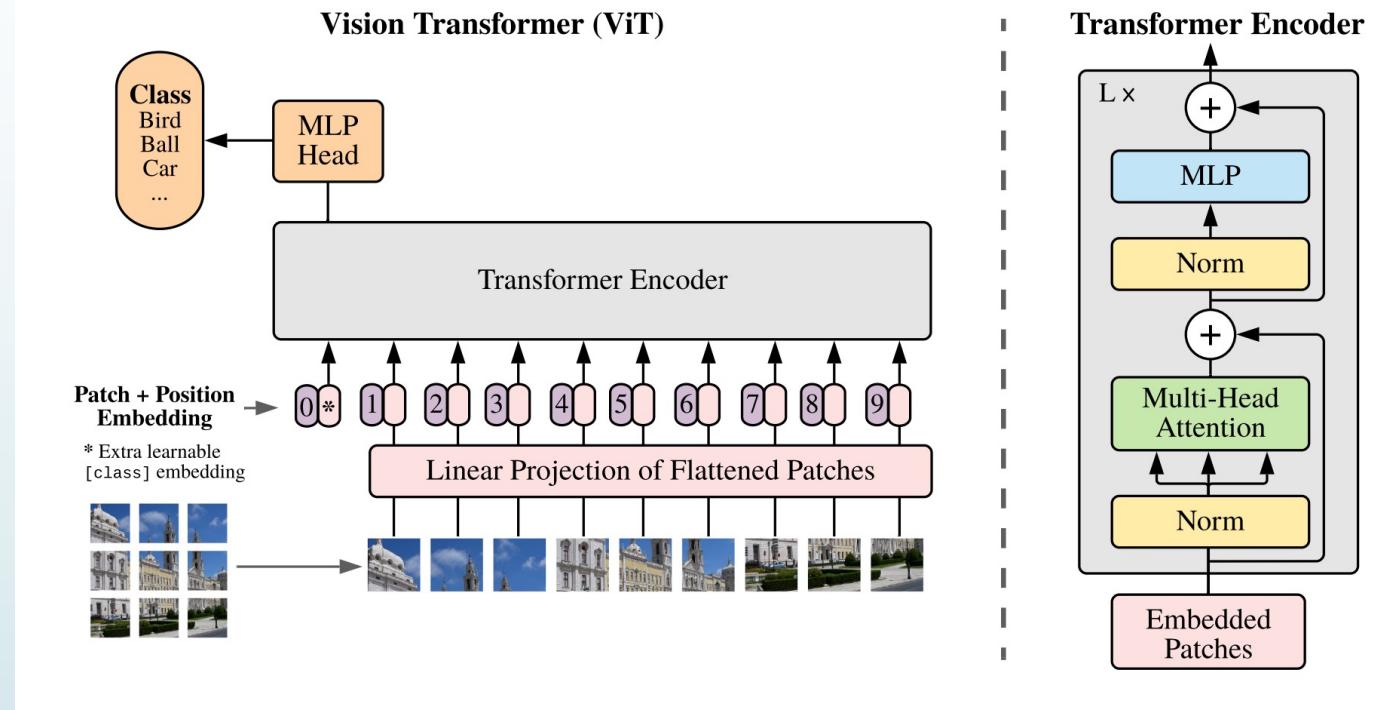


Fig. 20. The self-attention mechanism in SAGAN. (Image source: Fig. 2 in [Zhang et al., 2018](#))

Transformer de vision

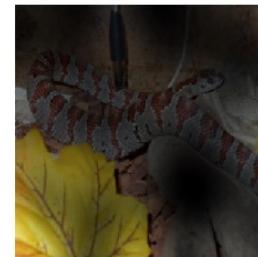
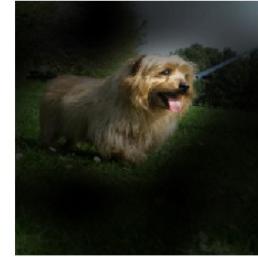
- ▶ Utilise des patchs !
- ▶ (réutilise les idées de l'histogramme des gradients)
- ▶ Attention interne entre les patchs.
- ▶ Architecture profonde



Exemple d'attention

Attention utilisée pour la tâche:

Input Attention



Exemple: Vision en Pratique pour les Sciences des Données

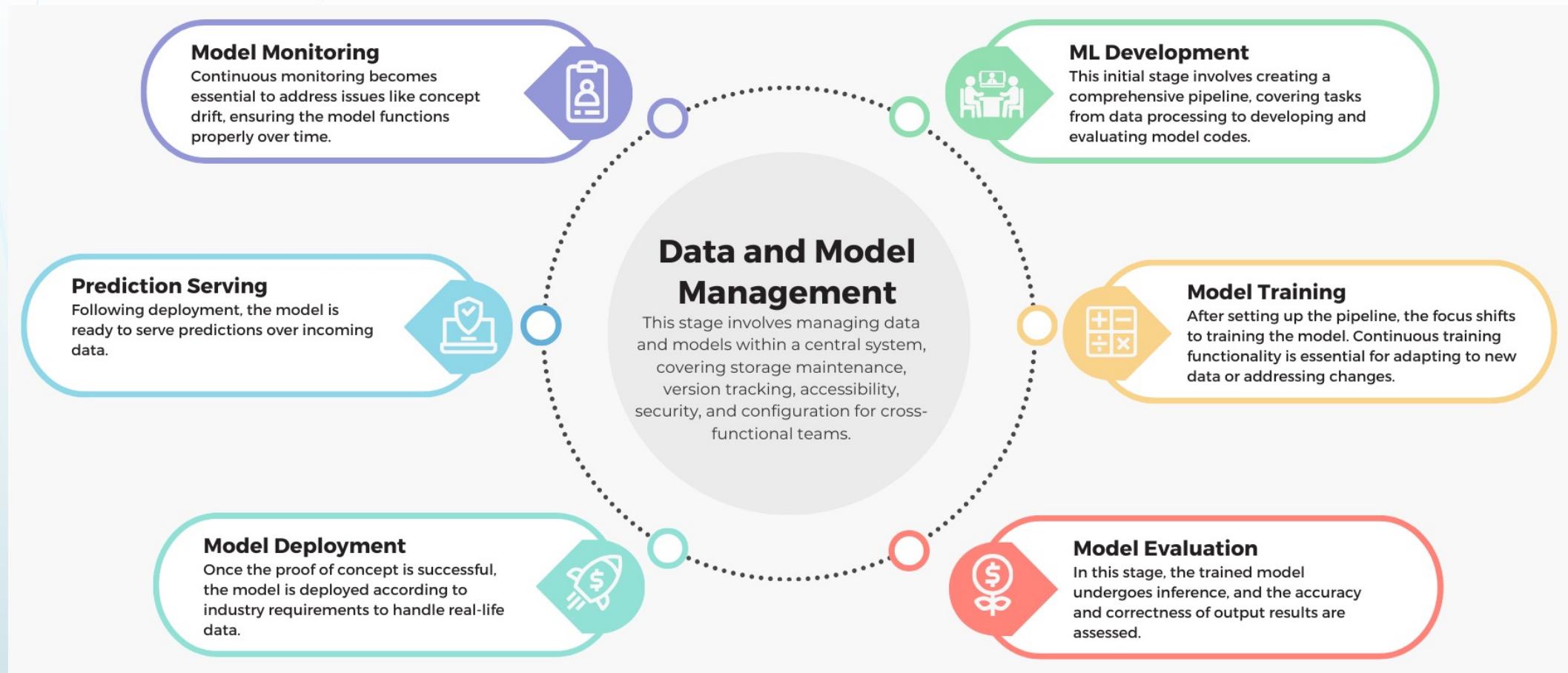
► Source:

<https://dagshub.com/blog/how-to-build-a-full-mlops-solution-for-computer-vision-using-oss-2/>

https://medium.com/@tenyks_blogger/computer-vision-is-already-evolving-3cd0e63e805b



La Vision dans le contexte de la Science des Données



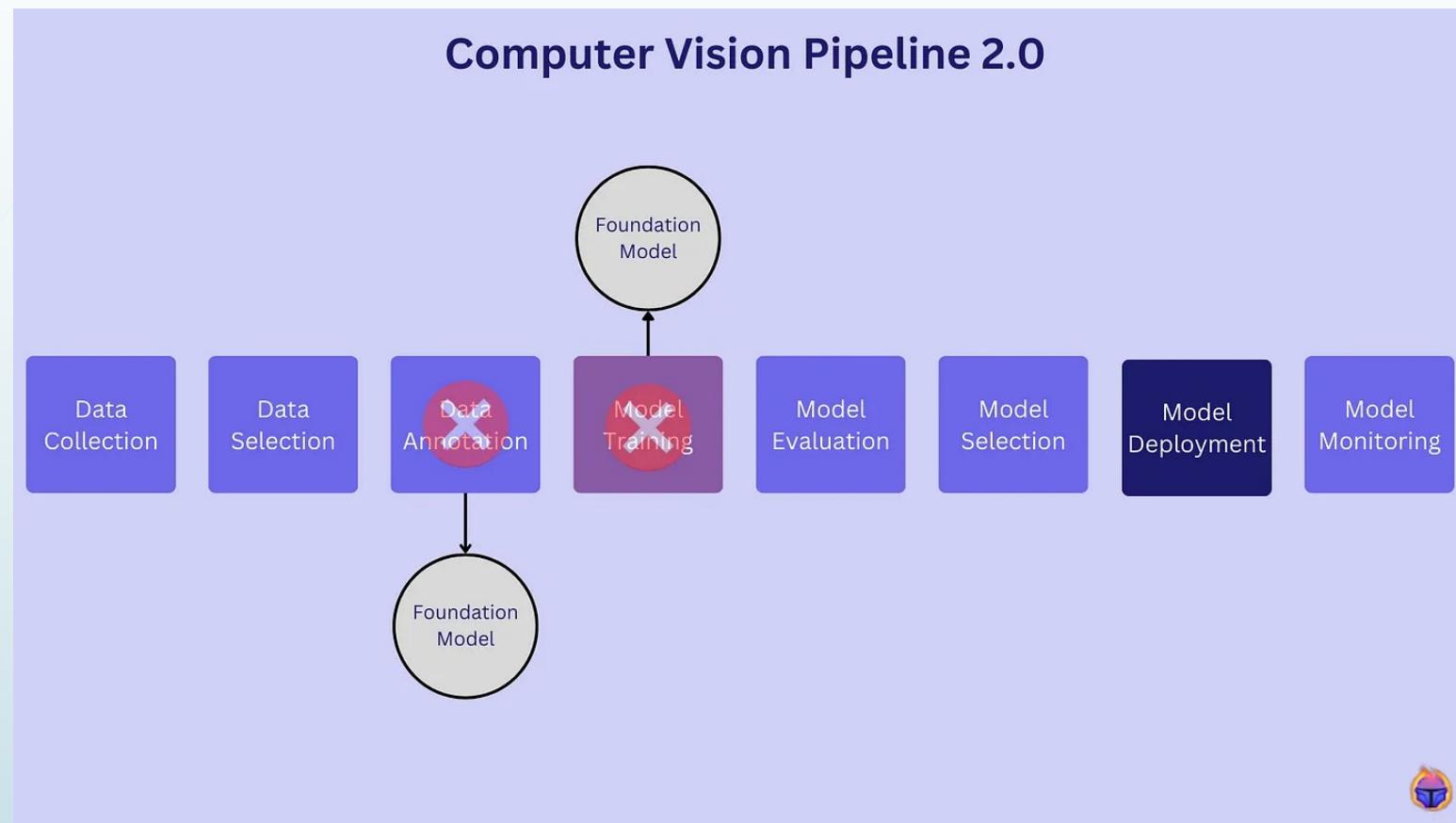
La Vision dans le contexte de la Science des Données

- ▶ Gestion des Données
 - ▶ La Vision par ordinateur requiert beaucoup de gestion des données
 - ▶ Collecte et étiquetage des données
 - ▶ De nombreuses entreprises ont développé des outils
 - ▶ Payer une entreprise pour étiqueter les données (par exemple [LabelStudio](#))



La Vision dans le contexte de la Science des Données

- Les modèles de fondation remplacent certaines des étapes:



Modèles de fondation dans la vision pour la science des données

- ▶ Les modèles de fondation remplacent l'étiquetage à la main.



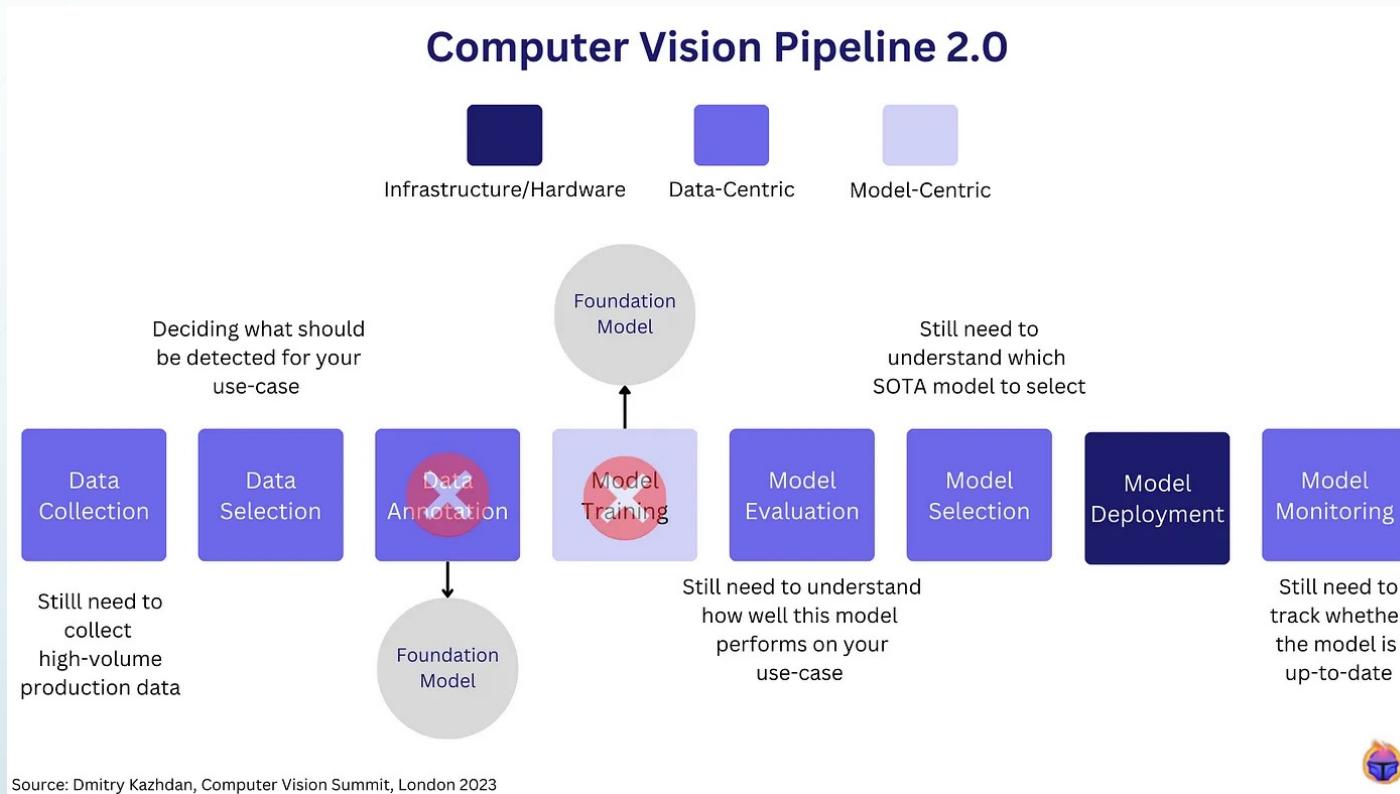
- ▶ This image captures a high-energy moment of equestrian barrel racing. The focus is on a horse and rider duo executing a tight turn around a barrel. The horse is a bay with a shiny coat, and its muscular build is on full display. It has its ears pinned back, showing concentration or the strain of the tight turn, and it wears a bridle with a bit and reins. The rider is leaning into the turn, showing good balance and control.

Modèles de fondation dans la vision pour la science des données

- ▶ Limitations
 - ▶ Les occultations sont difficiles
 - ▶ Les commandes (prompt) sont importantes.
 - ▶ Il faut donner les bonnes consignes
 - ▶ Le modèle est biaisé (très positif)
 - ▶ Les modèles sont gros
 - ▶ Lent et cher



Modèles de fondation dans la vision pour la science des données



Conferences sur la Vision

- ▶ CVPR : IEEE/CVF Conference on Computer Vision and Pattern Recognition
<http://cvpr2020.thecvf.com/>
- ▶ ICCV : IEEE/CVF International Conference on Computer Vision
<https://iccv2019.thecvf.com/>
- ▶ ACMMM : ACM International Conference on Multimedia
<https://2020.acmmm.org/>
- ▶ La vision est un sujet majeur des conferences d'apprentissage automatique: AAAI, IJCAI, ICML, NEURIPS, ...