

# Convolutional Neural Networks

(Apprentissage de représentation d'images)

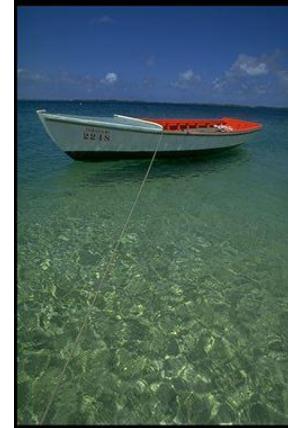
Stephane Ayache



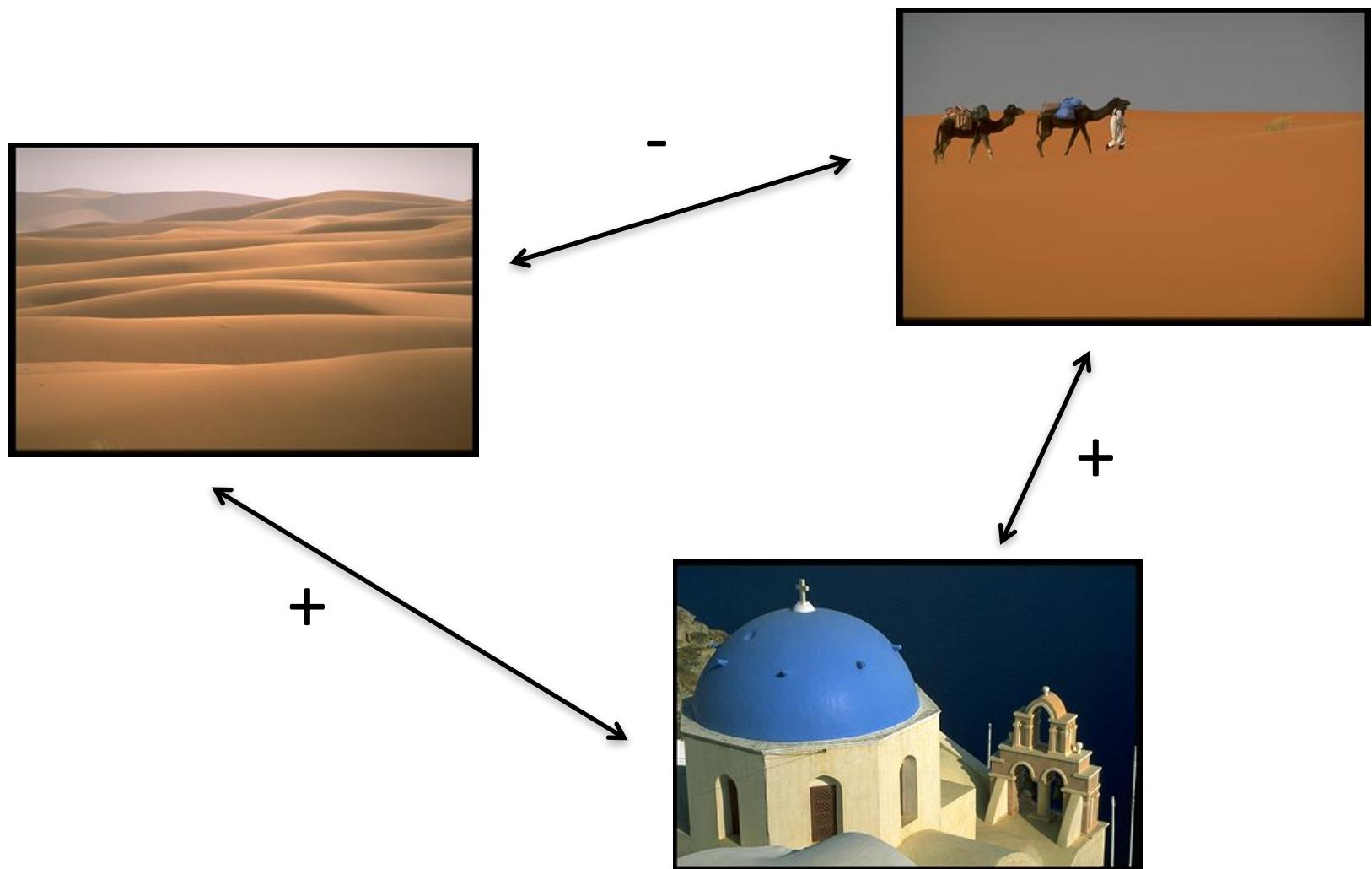
# Motivations



# Motivations



# Représentation et correspondance

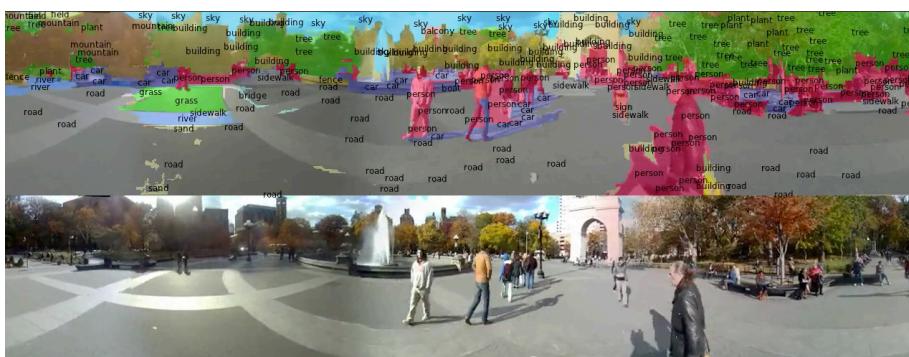
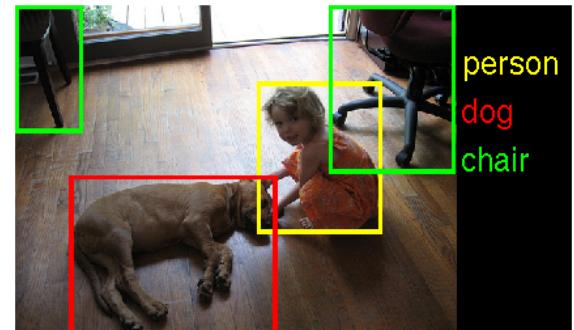
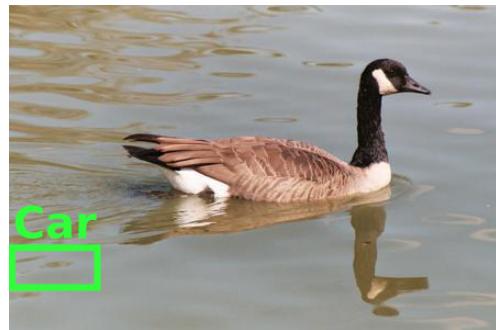


# Motivations

- Grande quantité d'images, de multiples applications
  - généraliste, médicale, robotique, biométrie, surveillance, ...
  - classification, visualisation, recherche d'information
- Besoin de comparer des images
  - La comparaison s'opère à partir d'une représentation
  - Les pixels constituent une première représentation mais très peu robustes aux variations de luminosité, échelle, occlusion, rotation, ...

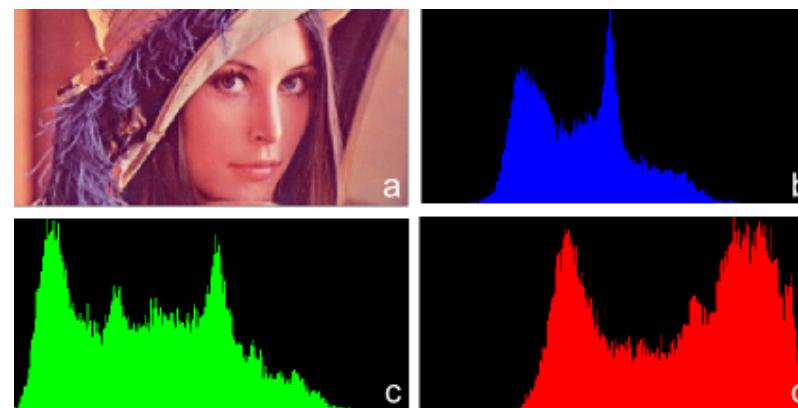
# Motivations

- Interprétation de scènes, détection d'objets
  - Plusieurs niveau d'interprétations, signal vs sémantique (cf cerveau humain)
  - Problème difficile !
- Très grand nombre d'images/vidéo disponibles, nombreuses classes
  - Les approches classiques ne passent pas à l'échelle
  - Les CNN ont permis des avancées majeures



# Représentation d'images

- Un histogramme par canal de couleur :
  - Histogramme composante par composante ou histogramme unique si l'image est monochrome
  - Discrétisation de l'espace des valeurs d'intensité, décomposé en intervalles complémentaires (« bins »)
  - Pour chaque intervalle, on calcule le pourcentage de pixels de l'image dont l'intensité s'y trouve



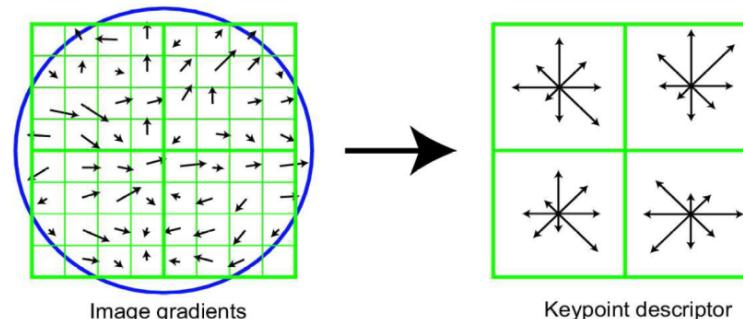
- Le vecteur de ces pourcentages est une représentation (descripteur d'image)

→ Similarité entre descripteurs  $\Leftrightarrow$  similarité entre images ?  
 $\Leftrightarrow$  similarité sémantique ?

# Représentation d'images par convolution

- Détection de points de « haute courbure » ou « coins »
  - Points géométriques « singuliers » dans l'image
- Extraits par divers filtres [Schmid 2000]
  - Calcul les dérivées spatiales à une échelle donnée
  - Convolution par des fonctions gaussiennes (Sobel, ...)
  - Construction d'invariants par une combinaison appropriée de ces diverses dérivées (module du gradient par exemple) : SIFT, SURF
- Représentation à partir de gradients spatiaux : HOG, SIFT, ...

➔ Etat de l'art avant les CNNs



# Filtrage d'image

- Le traitement d'un pixel dépend de ses voisins
- Permet de nombreuses opérations sur les images
  - Flou, points d'intérêt, contour, réhaussement, ...
- Espace Fourier / Spatial
  - Une multiplication dans le domaine de Fourier équivaut à une convolution dans le domaine spatial
- Opérateur de convolution (cas discret)

$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1, n_2] \cdot g[x - n_1, y - n_2]$$

# Filtrage : convolution

- Opérateur de convolution
  - Parcourt de l'image par une fenêtre glissante : le noyau de convolution
- Approximation dans un voisinage de pixels
  - Taille d'un filtre 2D : 3x3, 5x5, 7x7, ...
- Exemple en 1D

$$\begin{bmatrix} 1 & \mathbf{2} & 3 & 4 & 5 & 6 & 7 \\ 2 & 3 & 1 \end{bmatrix} \quad \rightarrow \quad 2 + 6 + 3 = \mathbf{11}$$

$$\begin{bmatrix} 1 & 2 & \mathbf{3} & 4 & 5 & 6 & 7 \\ 2 & 3 & 1 \end{bmatrix} \quad \rightarrow \quad 4 + 9 + 4 = \mathbf{14}$$

$$\begin{bmatrix} 1 & 2 & 3 & \mathbf{4} & 5 & 6 & 7 \\ 2 & 3 & 1 \end{bmatrix} \quad \rightarrow \quad 6 + 12 + 5 = \mathbf{23}$$

La réponse de la convolution est le vecteur : [11 14 23 29 35]

# Quelques filtres connus

- Filtre « Moyenneur »

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

- Alternative plus précise

$$\frac{1}{16} \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

- Intègre plus de 80% des approximations discrètes d'une gaussienne
- Génère du « flou » → diminue le contraste

# Détection d'orientations

- Noyaux des gradients horizontaux et verticaux :

Filtres de Sobel

$S_x$

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

$S_y$

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

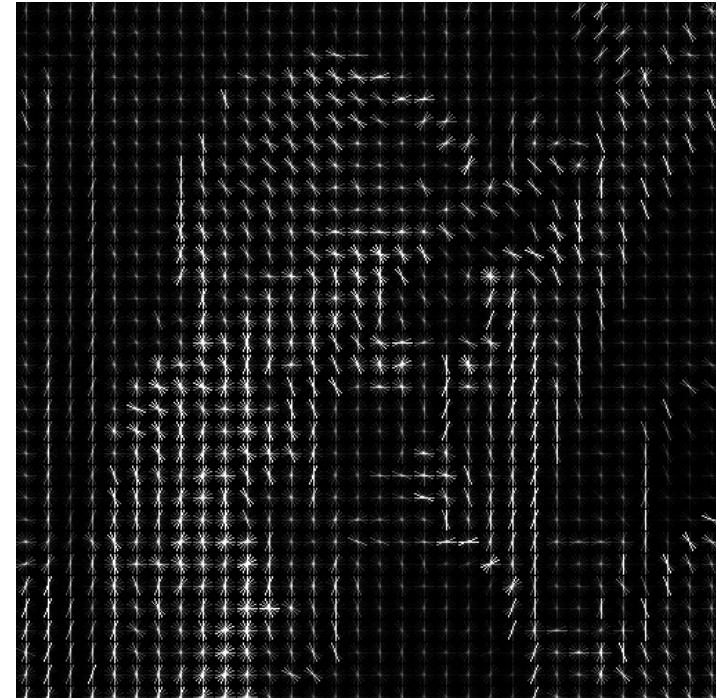
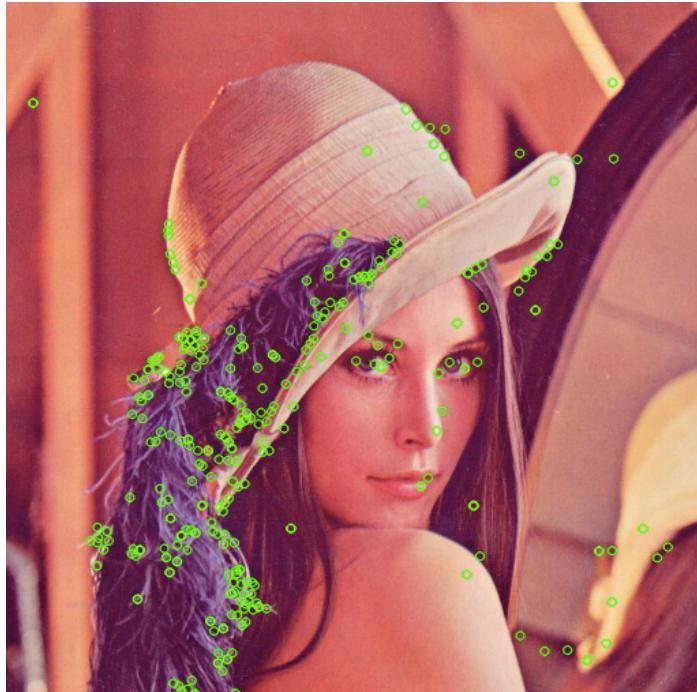
- Convolution par ces noyaux → détection des orientations



- Un descripteur possible : Histogramme des directions

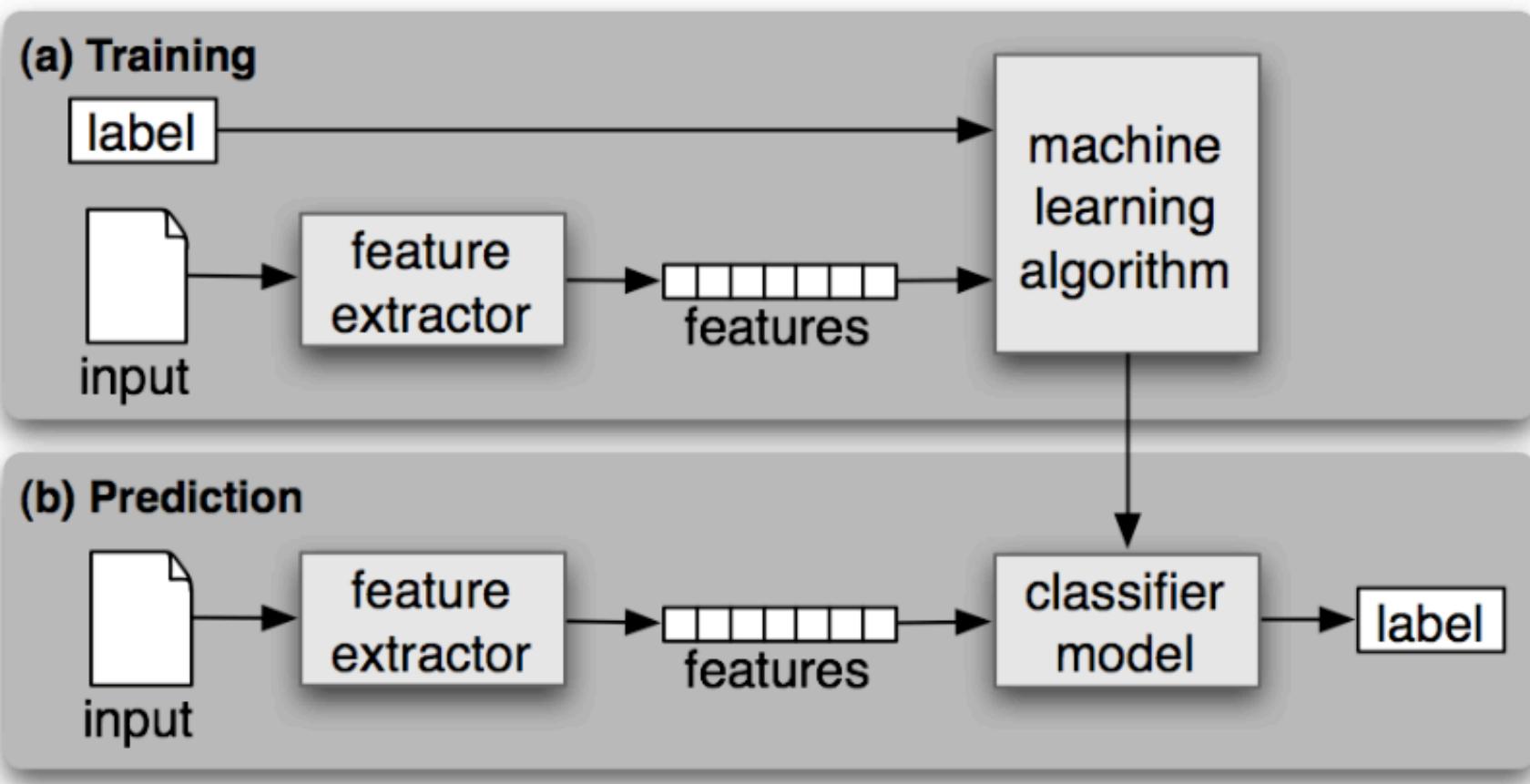
# Histogramme des gradients orientés (HOG) [Dalal, 2005]

- Norme du gradient :  $G = \sqrt{G_x^2 + G_y^2}$
- Direction du gradient :  $\Theta = \arctan\left(\frac{G_y}{G_x}\right)$
- Seuil sur la norme  $\Rightarrow$  détection de points d'intérêts



# Classification d'images "classique"

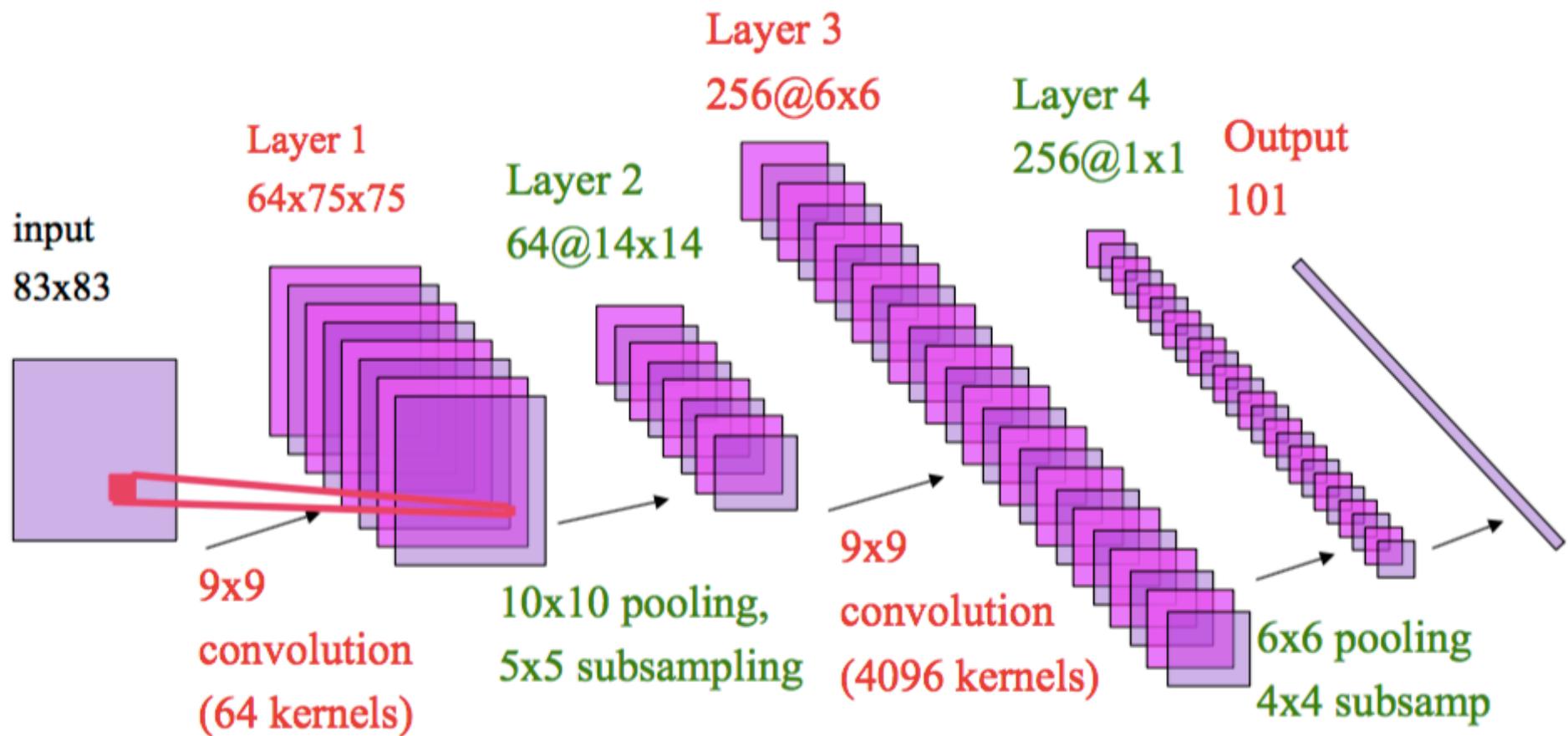
- Architectures "plates" (ou shallow)



# Apprentissage de représentation d'images

*Apprend des filtres spécifiques à une tache de classification ..*

# ConvNets

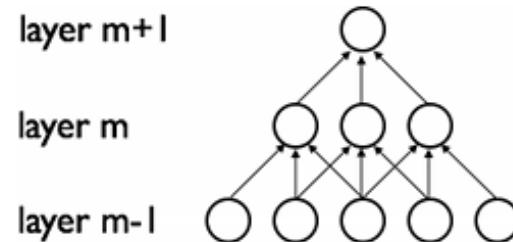


[Yann LeCun, 1998]

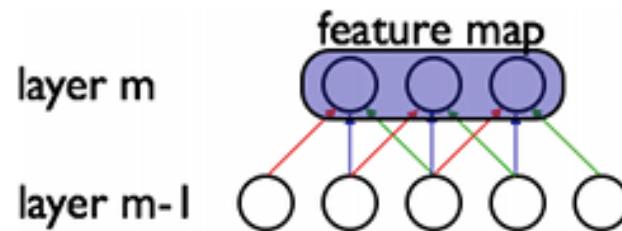
# Réseaux de neurones convolutionnels

- Un CNN est un DNN classique avec 2 propriétés supplémentaires

- Connexions locales



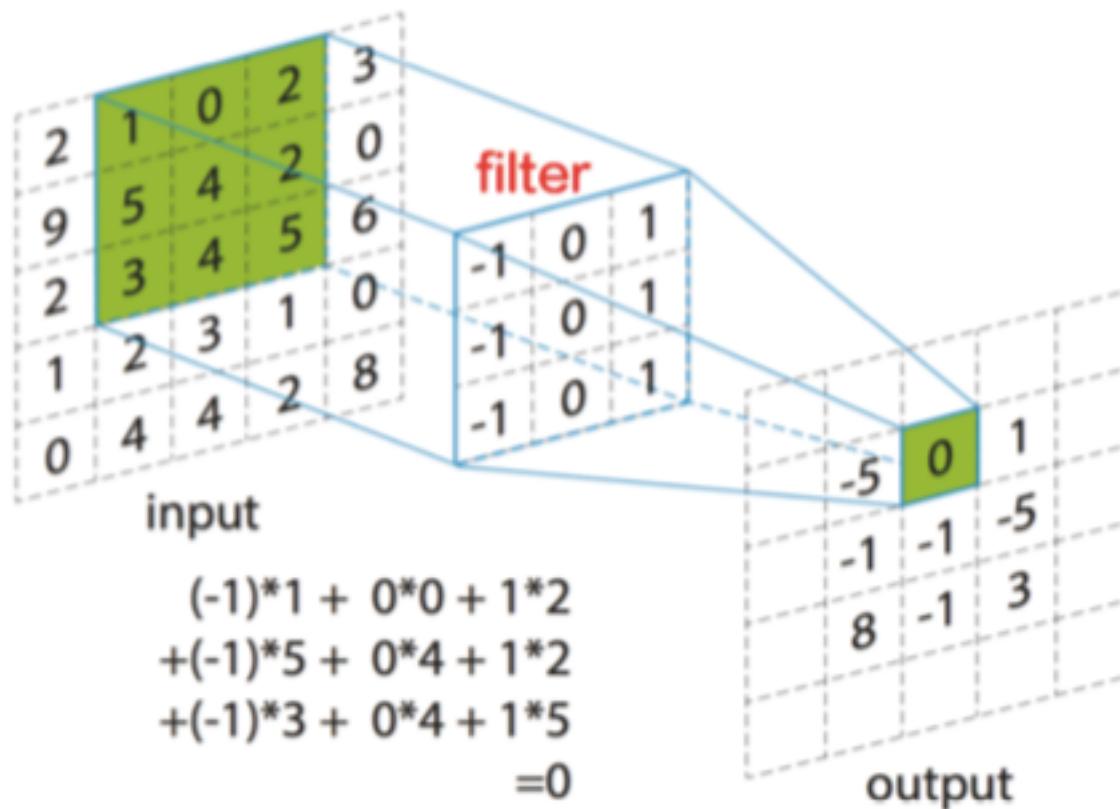
- Partage des poids entre couches

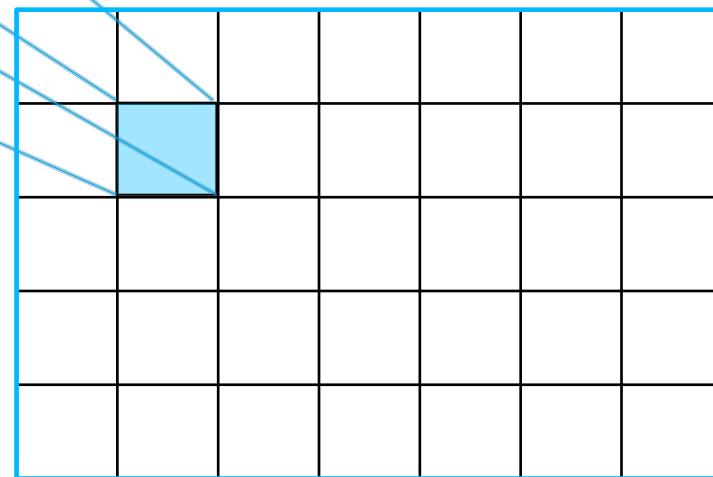
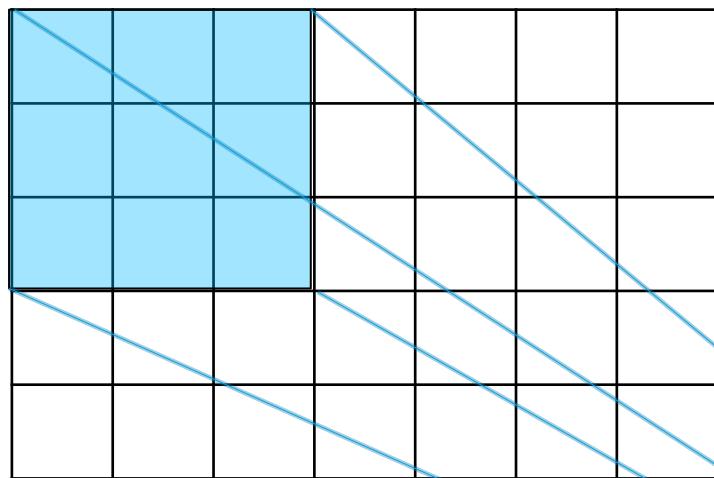


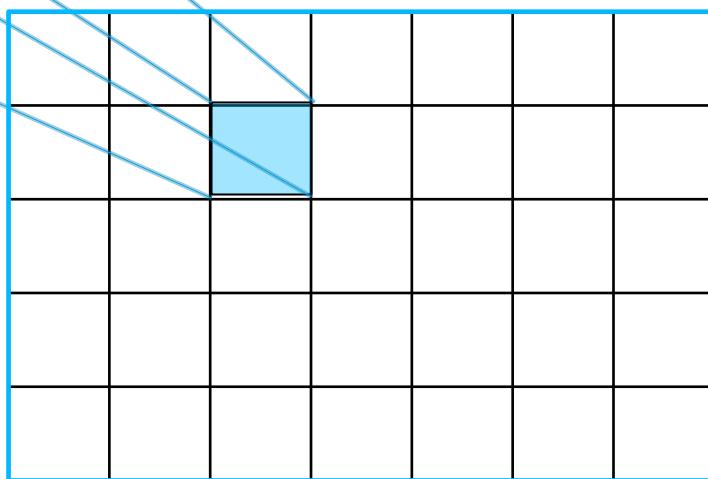
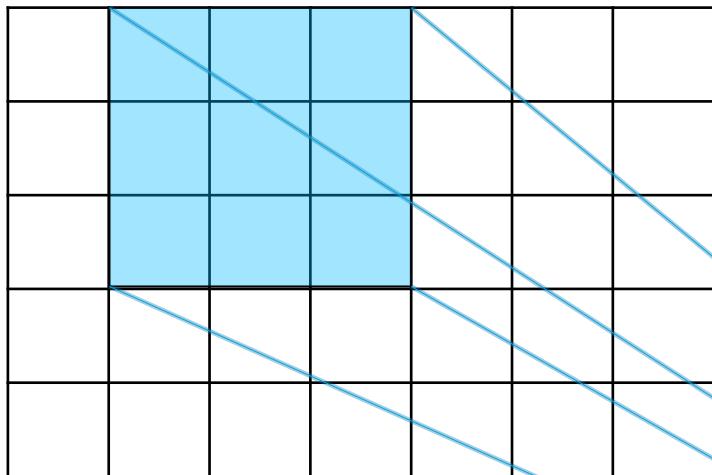
- Réduit drastiquement le nombre de paramètres

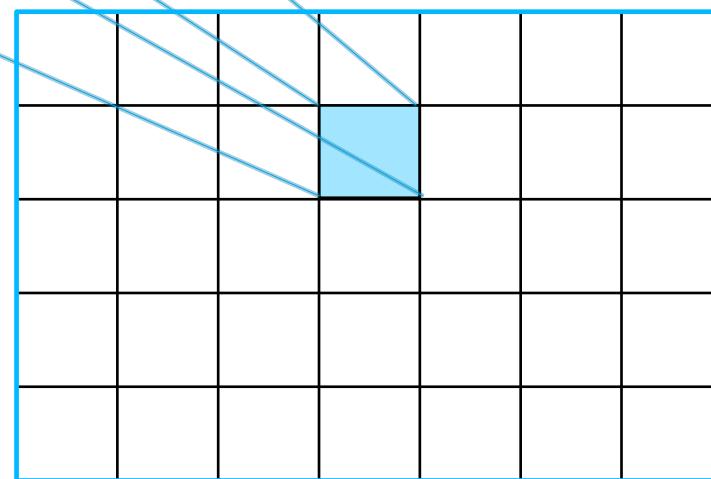
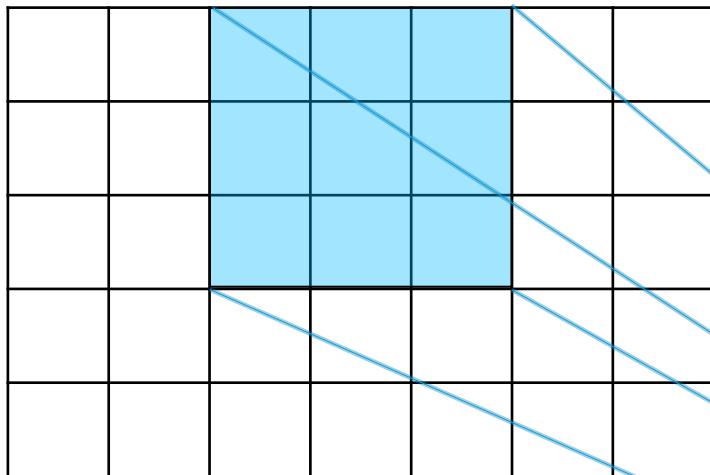
# Convolution 2D

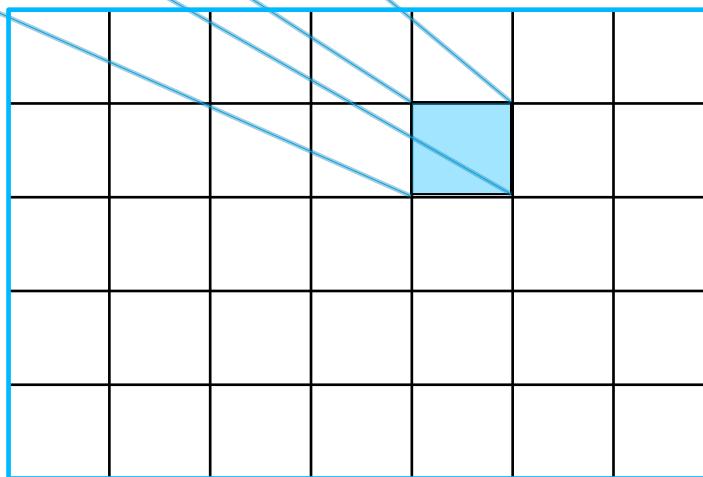
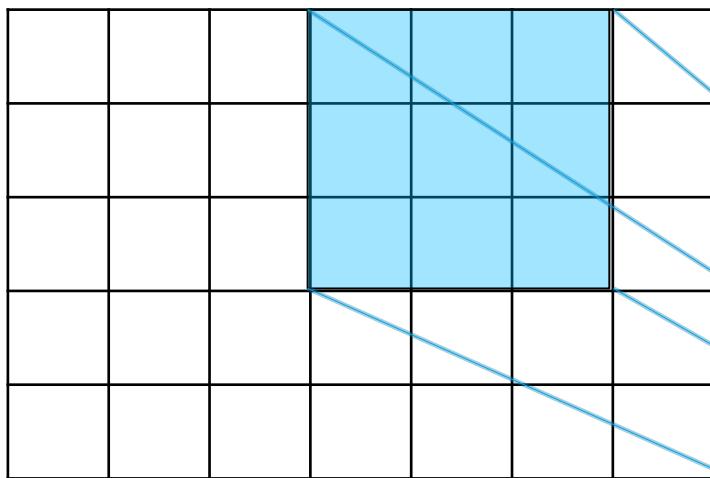
- Les poids d'une couche de convolution correspondent aux filtres (apris)
- Les activations de la couche sont en 2D, soit la réponse à la convolution (= feature map)

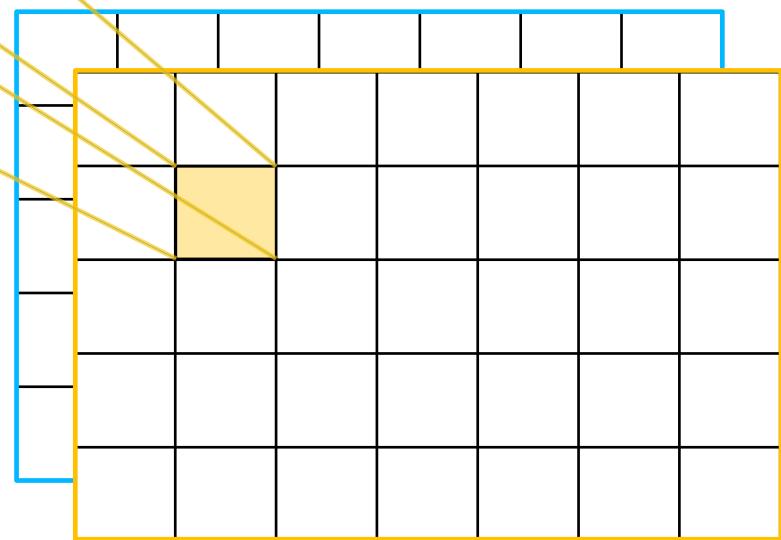
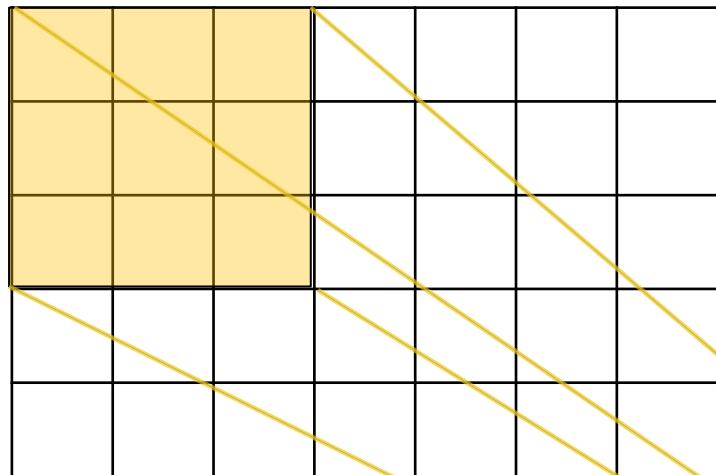


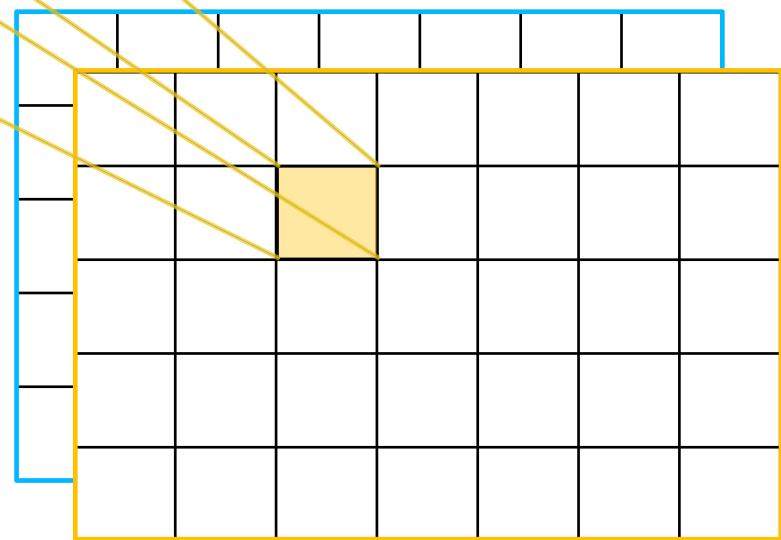
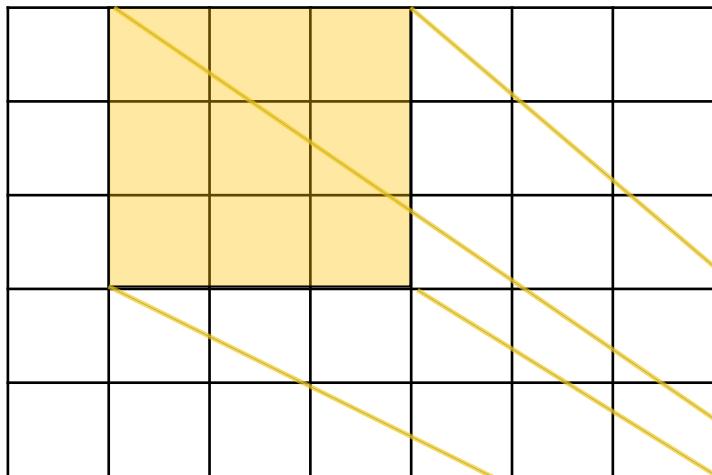


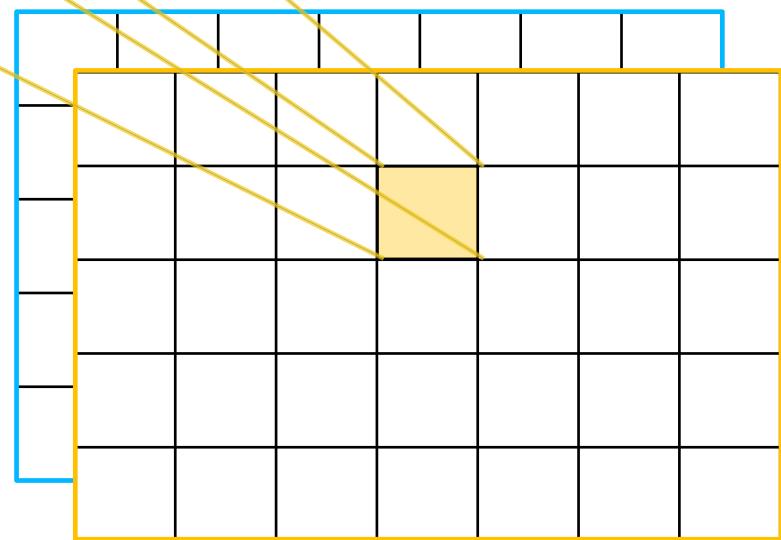
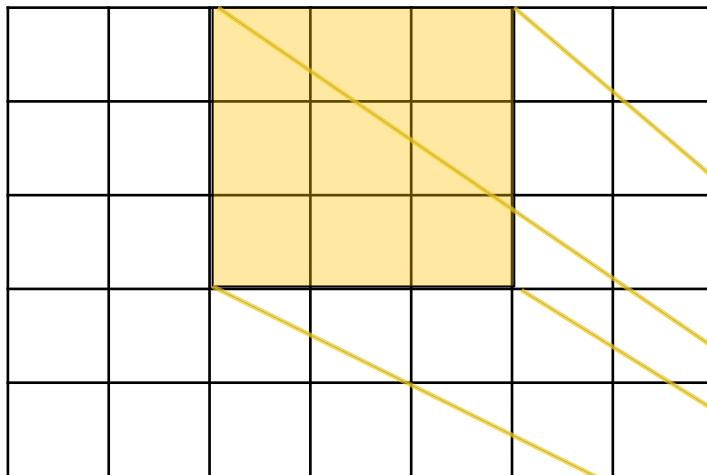


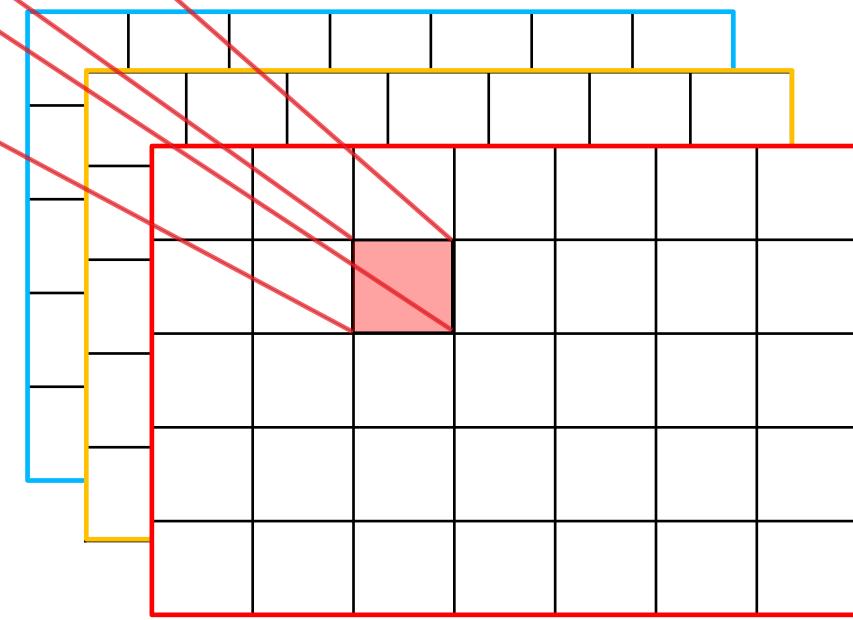
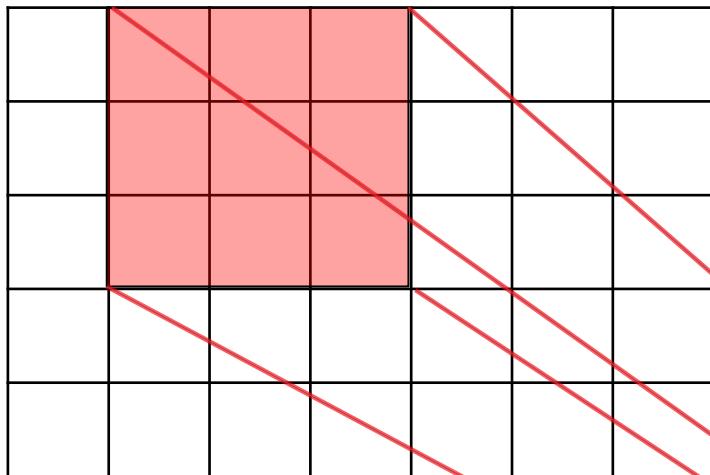


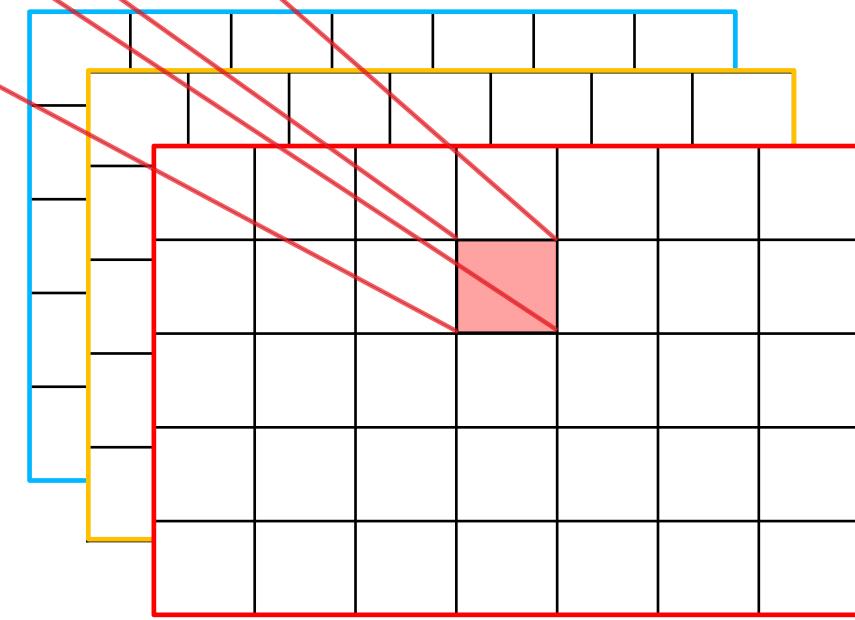
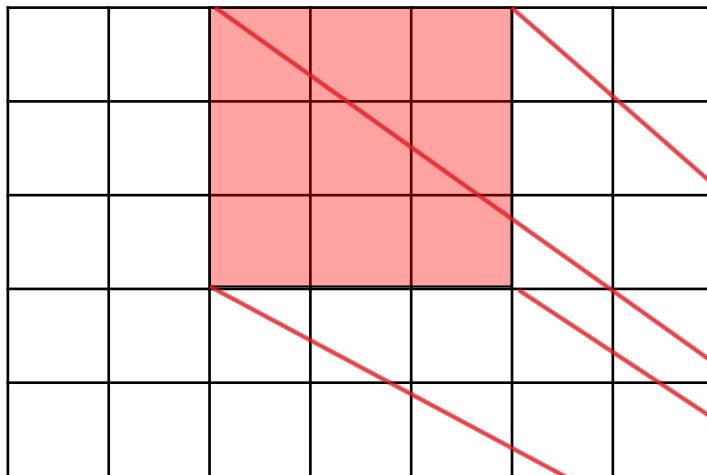




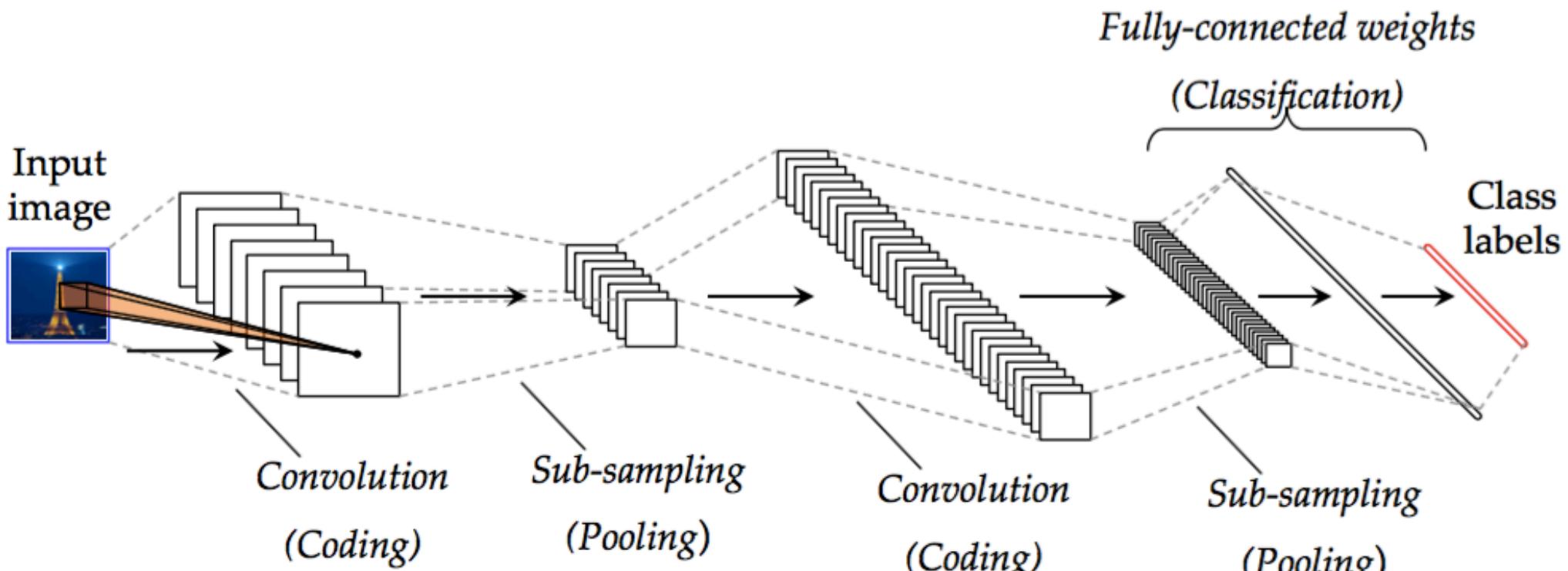






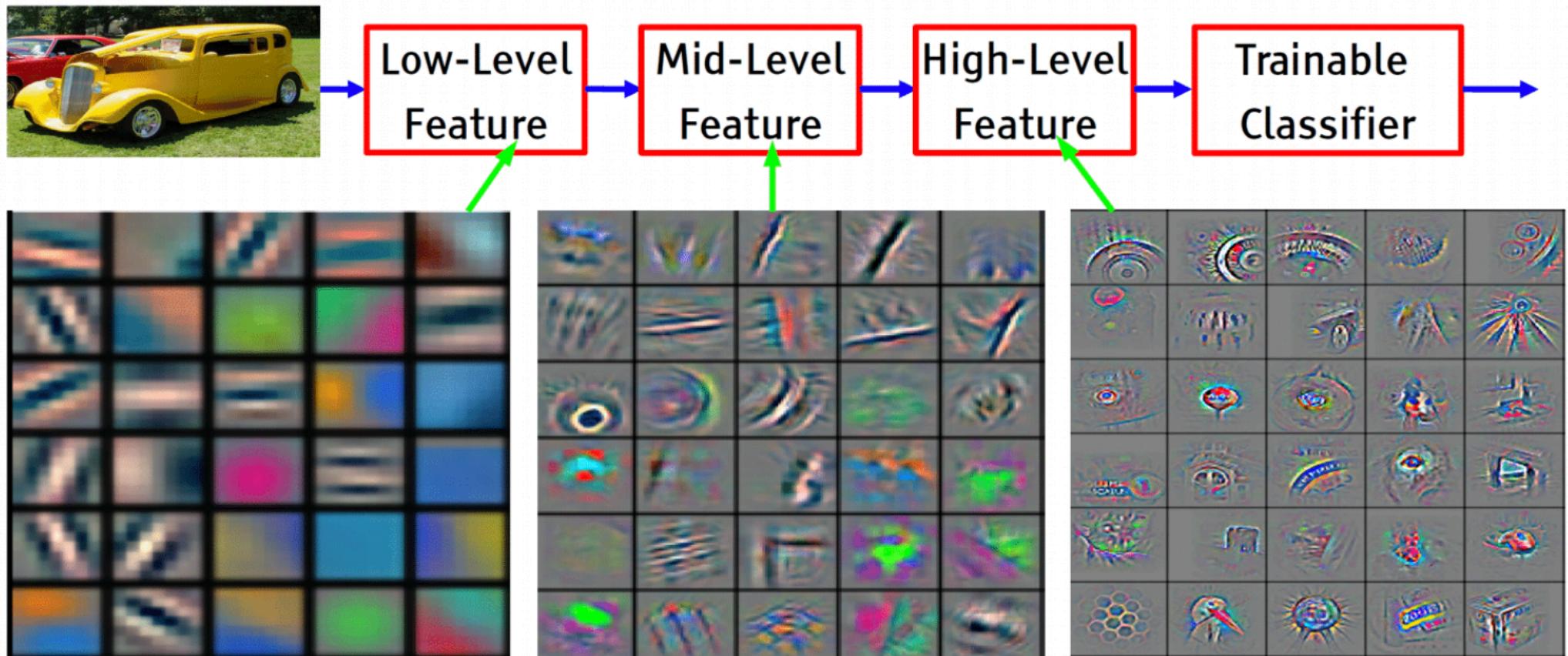


# Réseaux de neurones convolutionnels



Apprentissage de représentations

# Représentation hiérarchique



# Padding

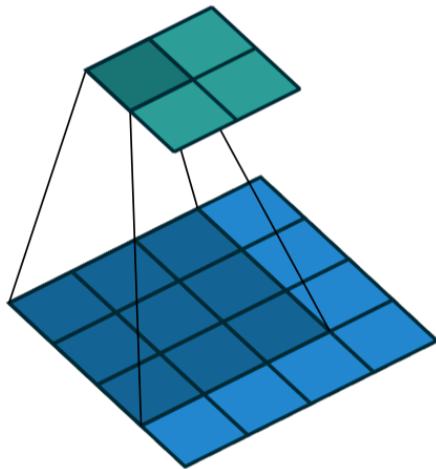
- Problème : convolution impossible pour les pixels en bordure
- Solution : Agrandir l'image initiale
  - ➔ Plusieurs approches possibles, ie: ajout de zéros
  - ➔ Le nombre de ligne/colonne à ajouter dépend de la taille du filtre
  - ➔ La taille du feature map dépend du padding

0 <sub>0</sub>	0 <sub>1</sub>	0 <sub>2</sub>	0	0	0	0
0 <sub>2</sub>	3 <sub>2</sub>	3 <sub>0</sub>	2	1	0	0
0 <sub>0</sub>	0 <sub>1</sub>	0 <sub>2</sub>	1	3	1	0
0	3	1	2	2	3	0
0	2	0	0	2	2	0
0	2	0	0	0	1	0
0	0	0	0	0	0	0

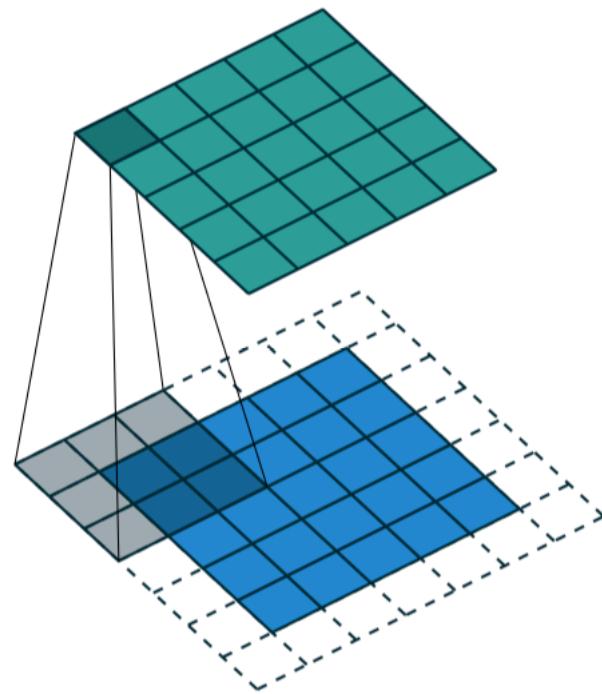
i = input size  
k = kernel size  
p = padding size  
o = output size

$$o = (i - k) + 2p + 1.$$

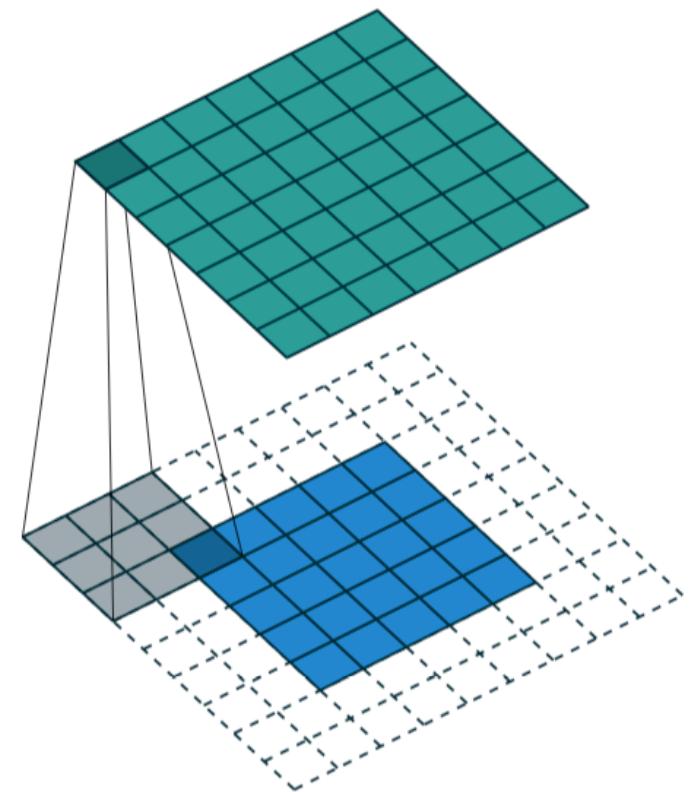
# Padding



No padding



Half padding



Full padding

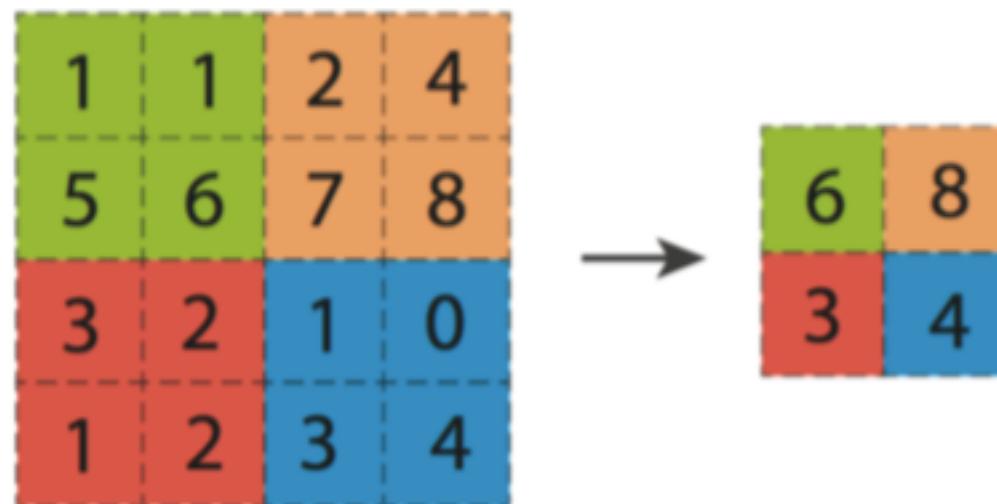
# Stride

- Pas (step) de convolution, en pixels
  - ➔ Généralement identique en w et h, pas nécessairement
  - ➔ Jusque là :  $s = 1$
- Si  $s > 1 \rightarrow$  réduit la taille des feature maps

$$o = \left\lfloor \frac{i + 2p - k}{s} \right\rfloor + 1$$

# Pooling

- Méthode d'agrégation locale
  - Réduit la taille des features map
  - Augmente l'invariance aux transformations (rotation, symétrie)
- Principalement 2 types de pooling
  - Average Pooling → conserve le plus d'information
  - Max Pooling → focus sur les forts contrastes

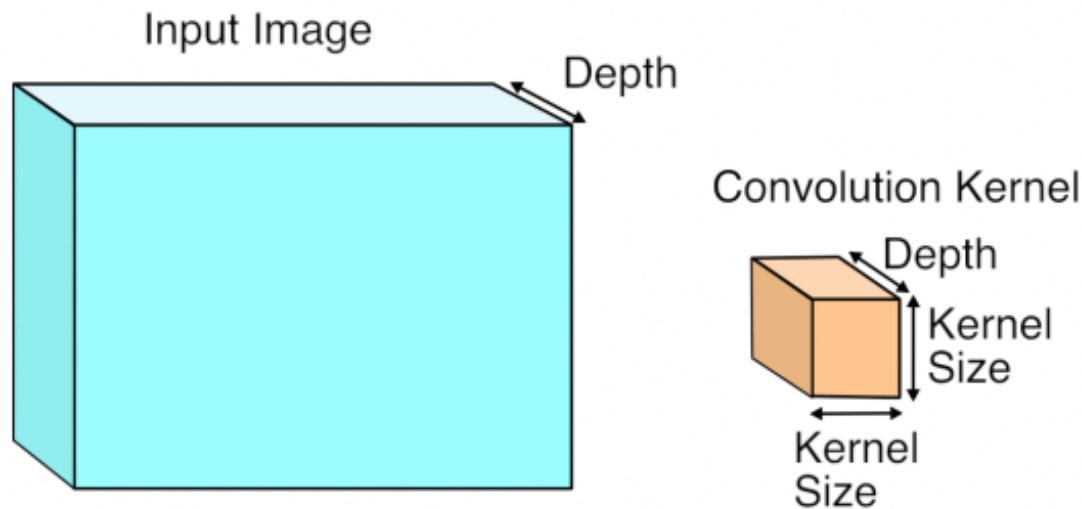


# Convolution 2D

- **Entrée** : Une image (multi-channels : C) de taille  $W \times H$
- **Paramètres** :
  1. Taille des filtres, ex :  $K \times K$
  2. Nombre de filtres, ex : F
  3. Stride : s = pas de la convolution
  4. Padding : méthode de complétion sur les bords
  5. + non-linéarité, biais, régularisation, etc
- Apprend F filtres de taille  $K \times K \times C$  (+ le biais = F)
- **Sortie** : F features maps de taille  $W \times H$  (si stride = 1 et half padding)

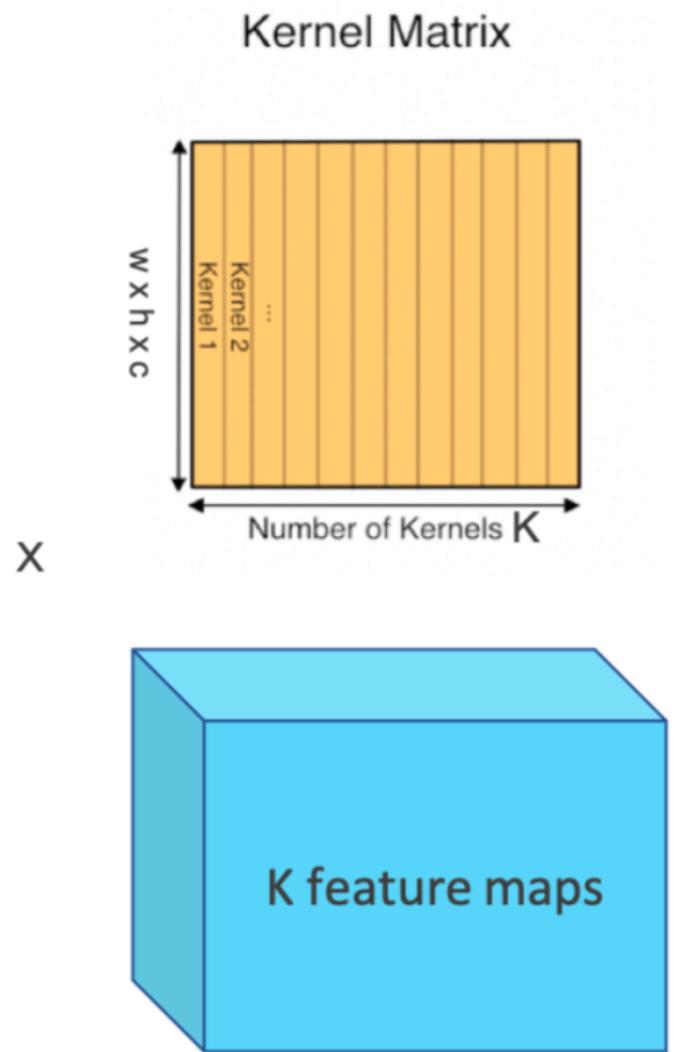
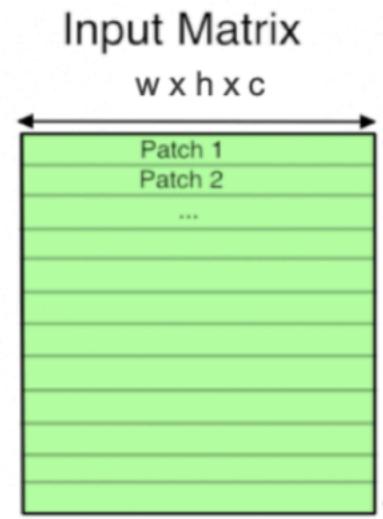
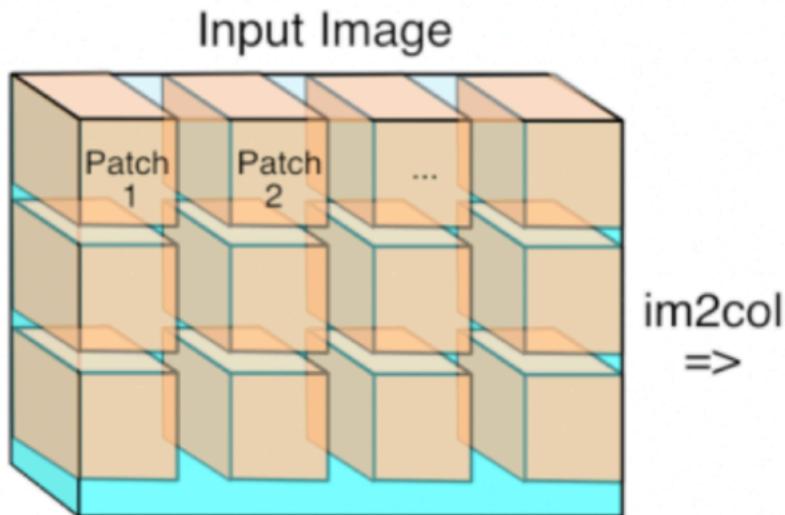
# Convolution 2D sur GPU

- Implémentation par produit matriciel
  - Duplication d'informations
  - Opération efficace sur GPU



# Convolution 2D sur GPU

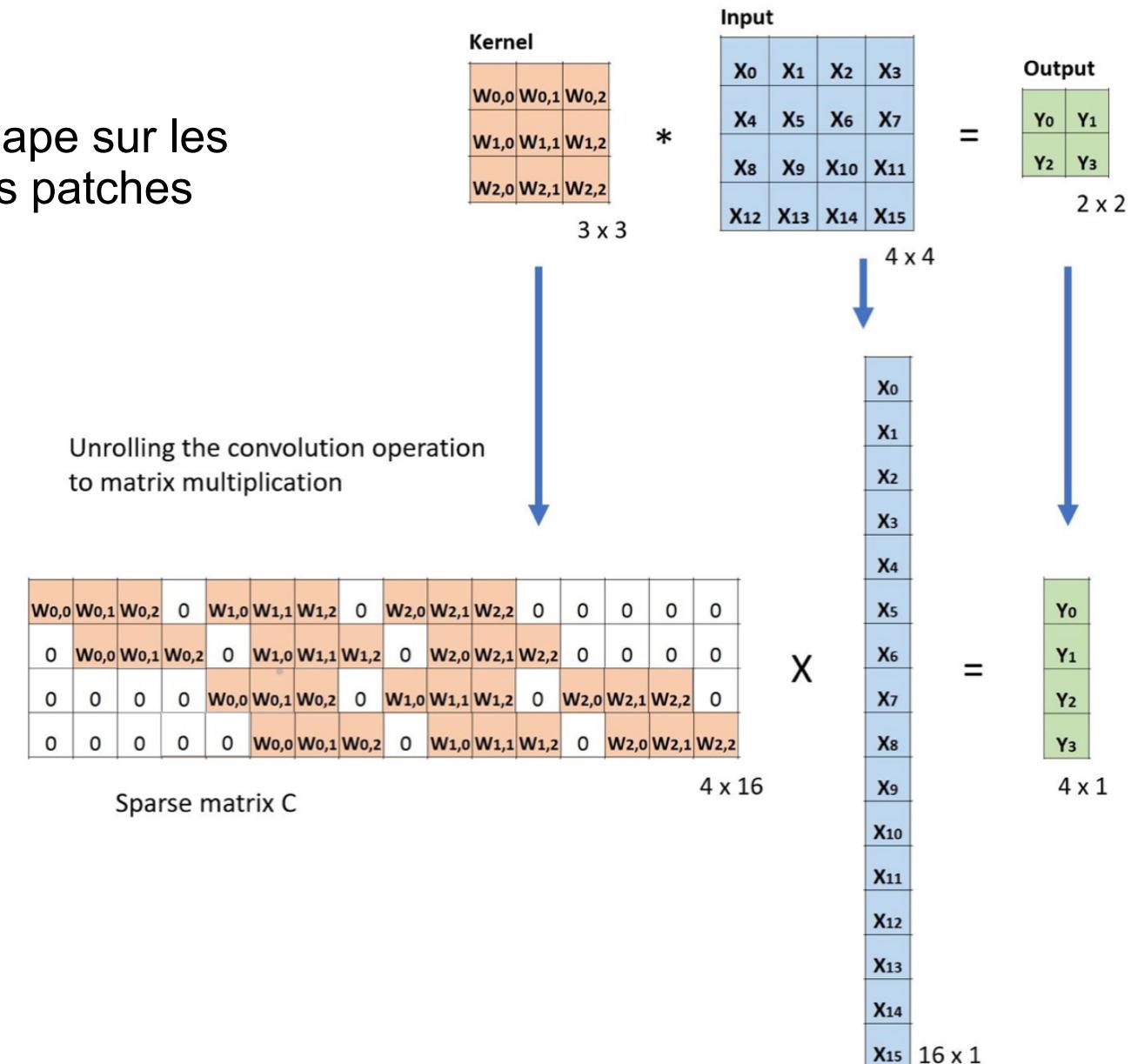
- Implémentation par produit matriciel
  - Duplication d'informations
  - Opération efficace sur GPU



# Convolution 2D sur GPU (2)

- Variante avec reshape sur les filtres plutôt que les patches

→ Matrice sparse



# Complexité

- Filtres de taille  $K \times K$
- Features map de taille  $H \times W$
- $C$  channels en entrée
- $F$  channels (filtres) en sortie

→ mémoire :  $K \times K \times C \times F \times H \times W$

→ Paramètres appris :  $K \times K \times K \times F (+ F)$

# Variantes de convolution

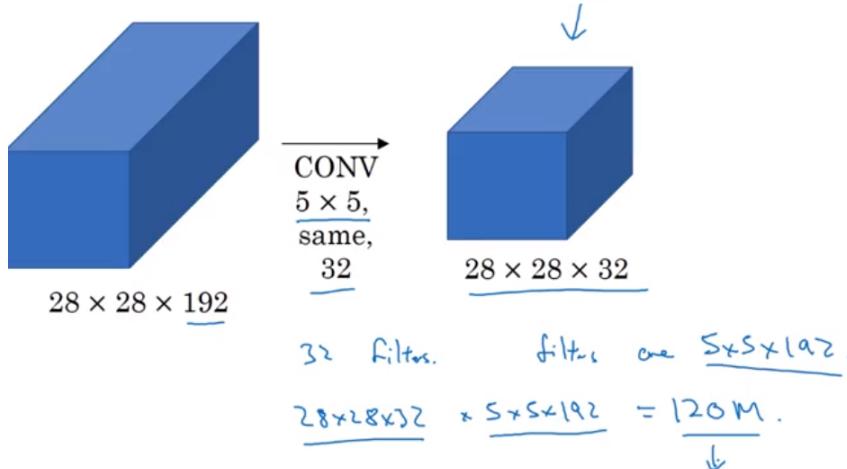
- 1D, 2D, 3D convolution
- $1 \times 1$  convolution
- Separable convolution
- Dilated convolution
- Transposed convolution

# 1x1 Convolution

- Filtres de taille 1x1 !
  - A utiliser après une couche de convolution
  - Un seul filtre : apprend les poids d'une combinaison linéaire entre les channels de la couche précédente
  - Plusieurs filtres : permettent une factorisation des features map dans l'espace des filtres, avec peu de paramètres
- Réduction des features map dans l'espace des filtres
- $C \times H \times W \rightarrow F \times H \times W$
  - avec seulement  $F$  paramètres

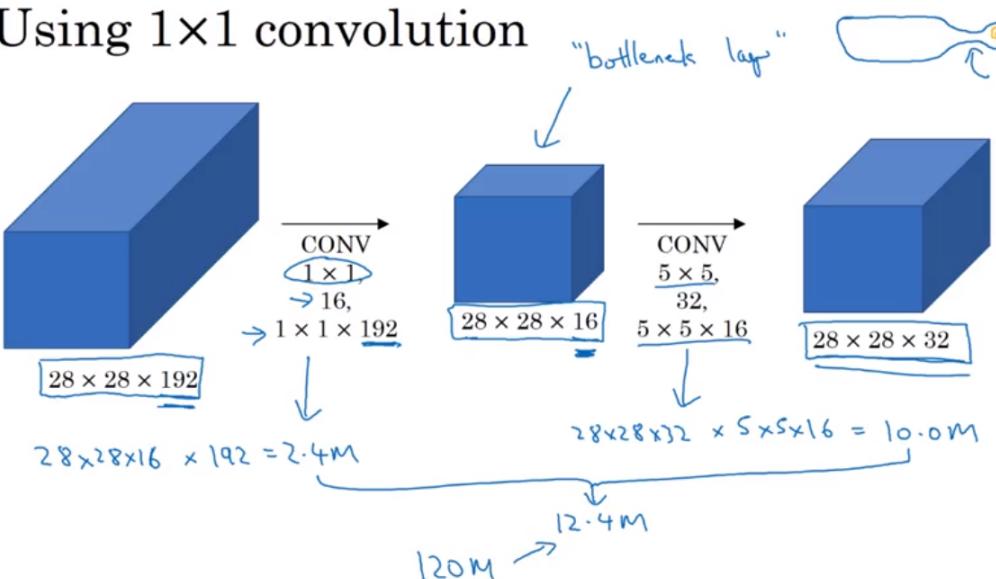
# 1x1 Convolution

The problem of computational cost



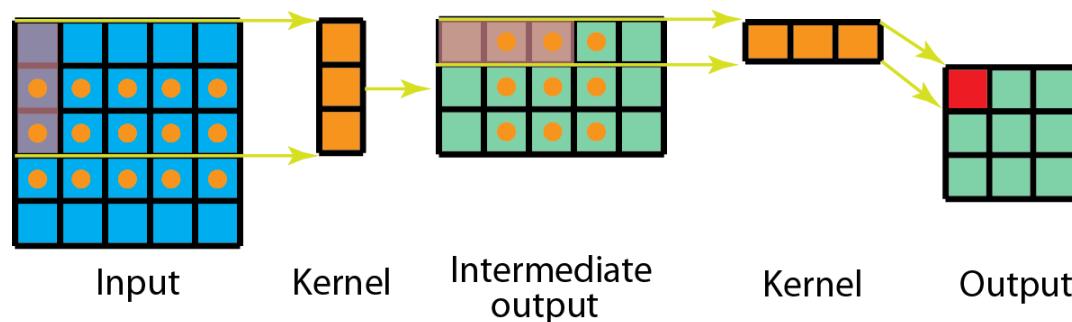
- Introduit dans l'architecture Inception (GoogleNet)
- <https://arxiv.org/abs/1409.4842>
- Réduit le nombre de channels avant d'opérer des convolutions de plus grands filtres
- Peut aussi ajouter une non linéarité supplémentaire

Using 1x1 convolution



# Separable convolution

- Introduit dans l'architecture MobileNet (Google)
- <https://arxiv.org/pdf/1704.04861.pdf>
- Réduit (factorisation) du nombre de paramètres à apprendre
  - Selon les dimensions spatiales : *Spatially separable convolution*

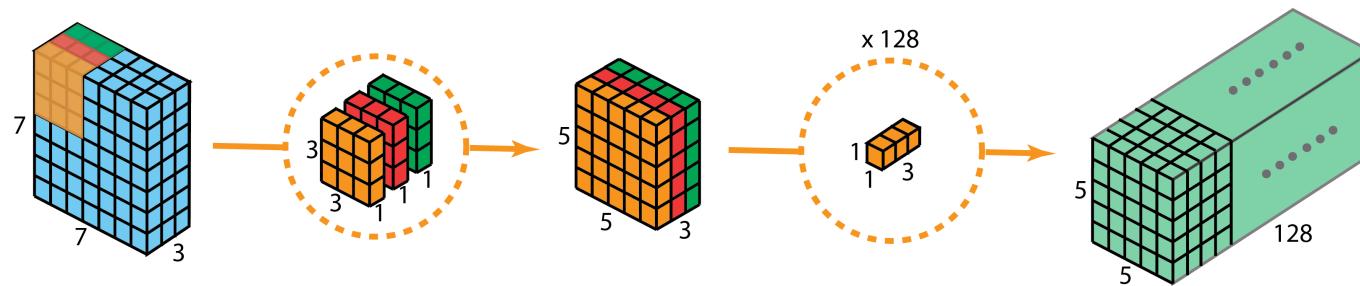


- Peu de gain, les filtres sont rarement très grands
- Tous les filtres ne sont pas de rank 1

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

# Separable convolution

- Introduit dans l'architecture MobileNet (Google)
- <https://arxiv.org/pdf/1704.04861.pdf>
- Réduit (factorisation) du nombre de paramètres à apprendre
  - Selon la profondeur : *Depthwise separable convolution*  
→ C convolutions séparées dans chaque channel d'entrée  
→ Puis F convolutions 1x1



# Dilated convolution

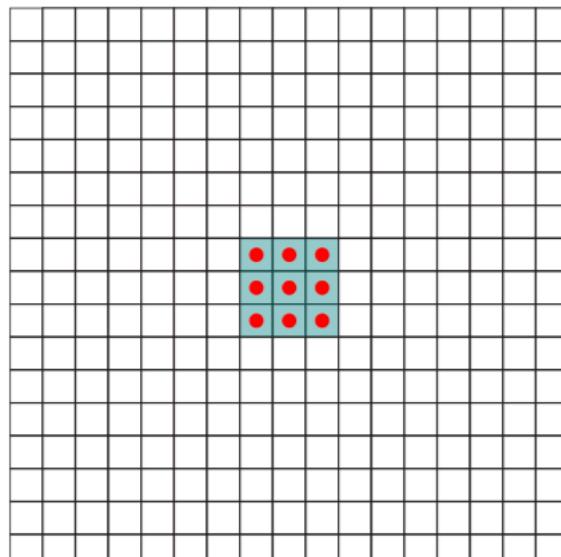
- Aussi appelé "Convolution a trou" (WaveNet, Segmentation, ...)
  - <https://arxiv.org/pdf/1609.03499.pdf>
  - <https://arxiv.org/pdf/1412.7062.pdf>
- Augmente la taille des filtres avec autant de paramètres

$$\hat{k} = k + (k - 1)(d - 1)$$

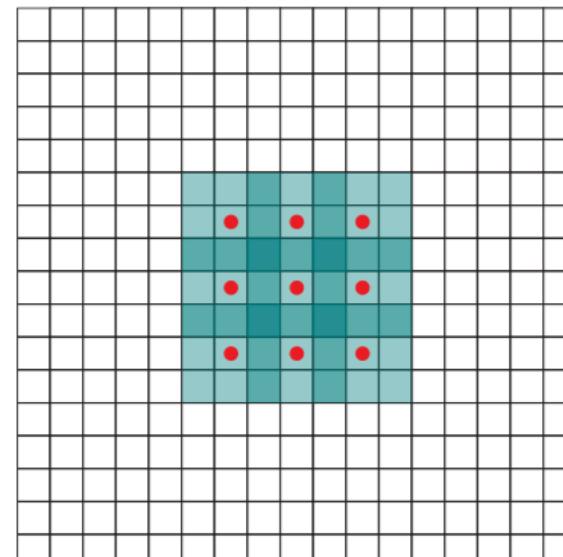
- Un filtre de taille  $k$ , dilaté d'un facteur  $d$ , est de taille
- Réduit la taille des features map 
$$o = \left\lfloor \frac{i + 2p - k - (k - 1)(d - 1)}{s} \right\rfloor + 1$$
- Mis en couches, augmente exponentiellement la taille des zones de pixels "vus" par un filtre (receptive fields)

# Dilated convolution

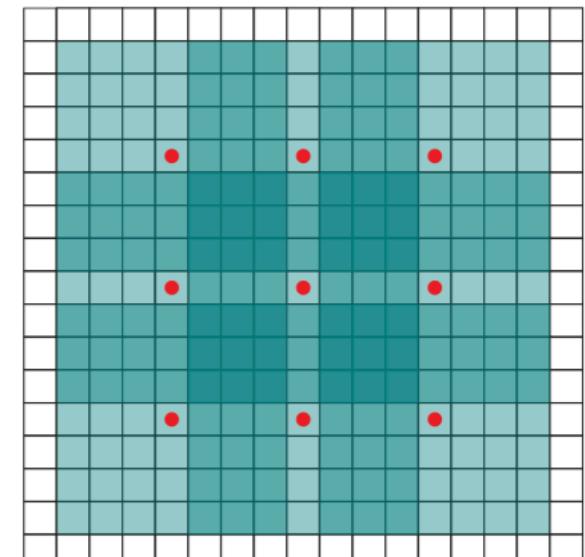
- Mis en couches, augmente exponentiellement la taille des zones de pixels "vus" par un filtre (receptive fields)



(a)



(b)

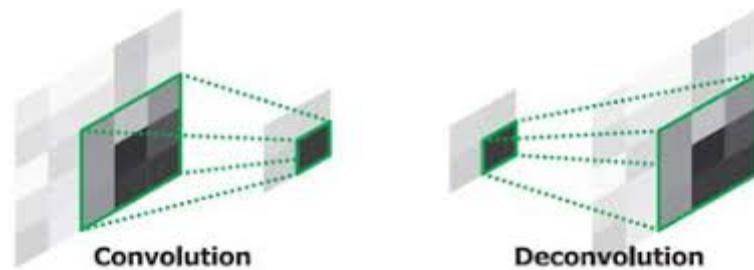


(c)

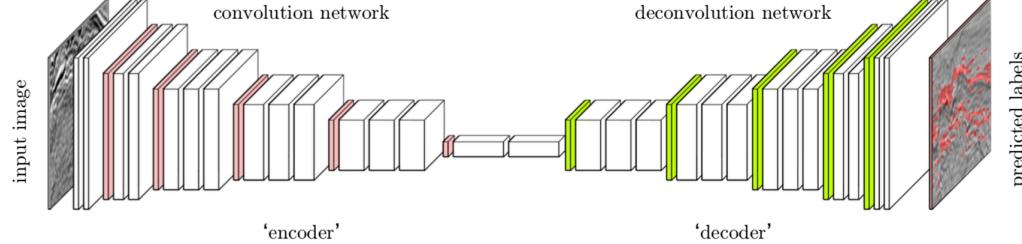
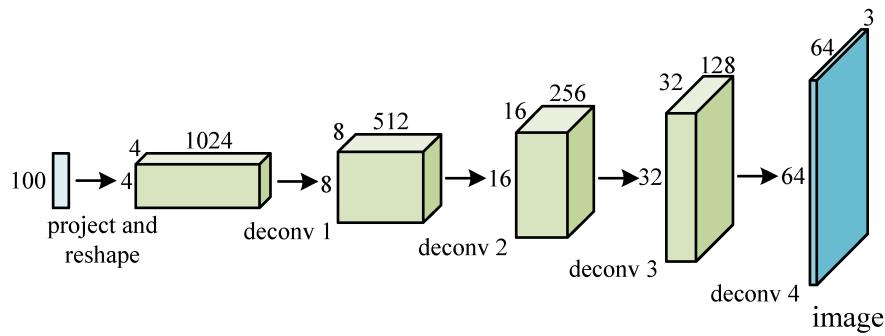
<https://arxiv.org/pdf/1511.07122.pdf> [Yu, ICLR 2016]

# Transposed Convolution

- Aussi appelé "Déconvolution"
- Features map plus grands que l'image d'entrée
- Alternative à Up-sampling + Convolution

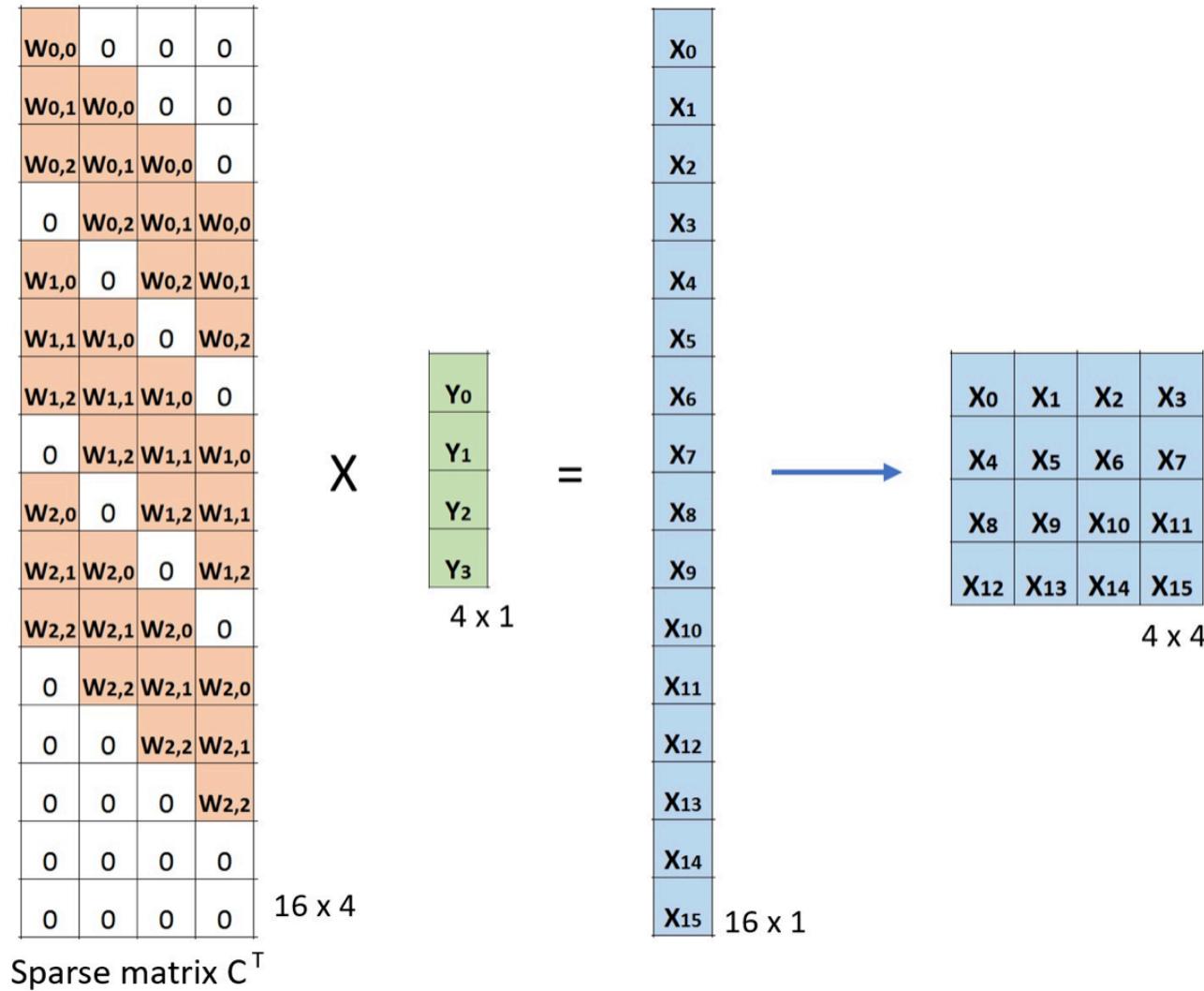


- Utilisé pour des modèles "générateurs"



# Transposed Convolution

- Pourquoi transposed ?

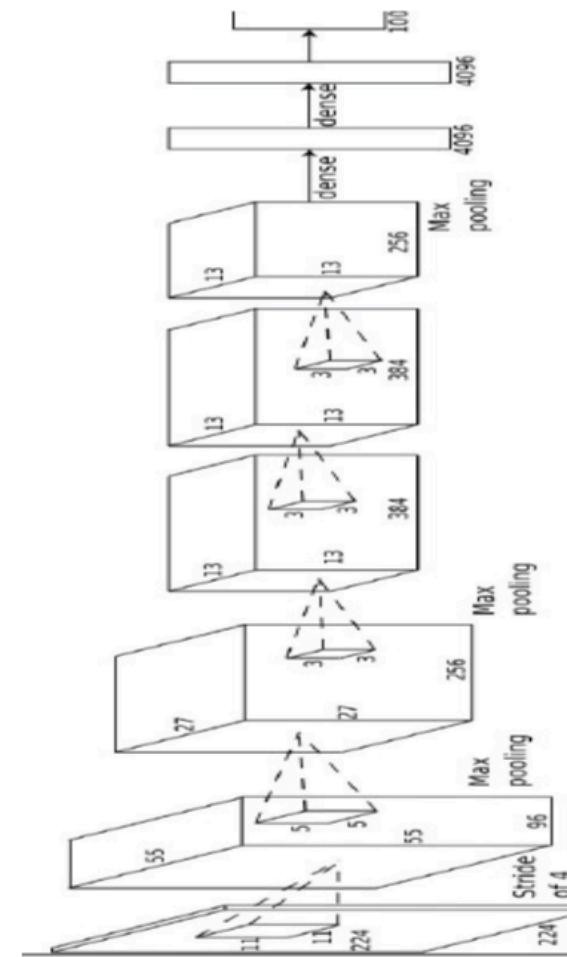


# Quelques exemples de CNN

# AlexNet – 2012

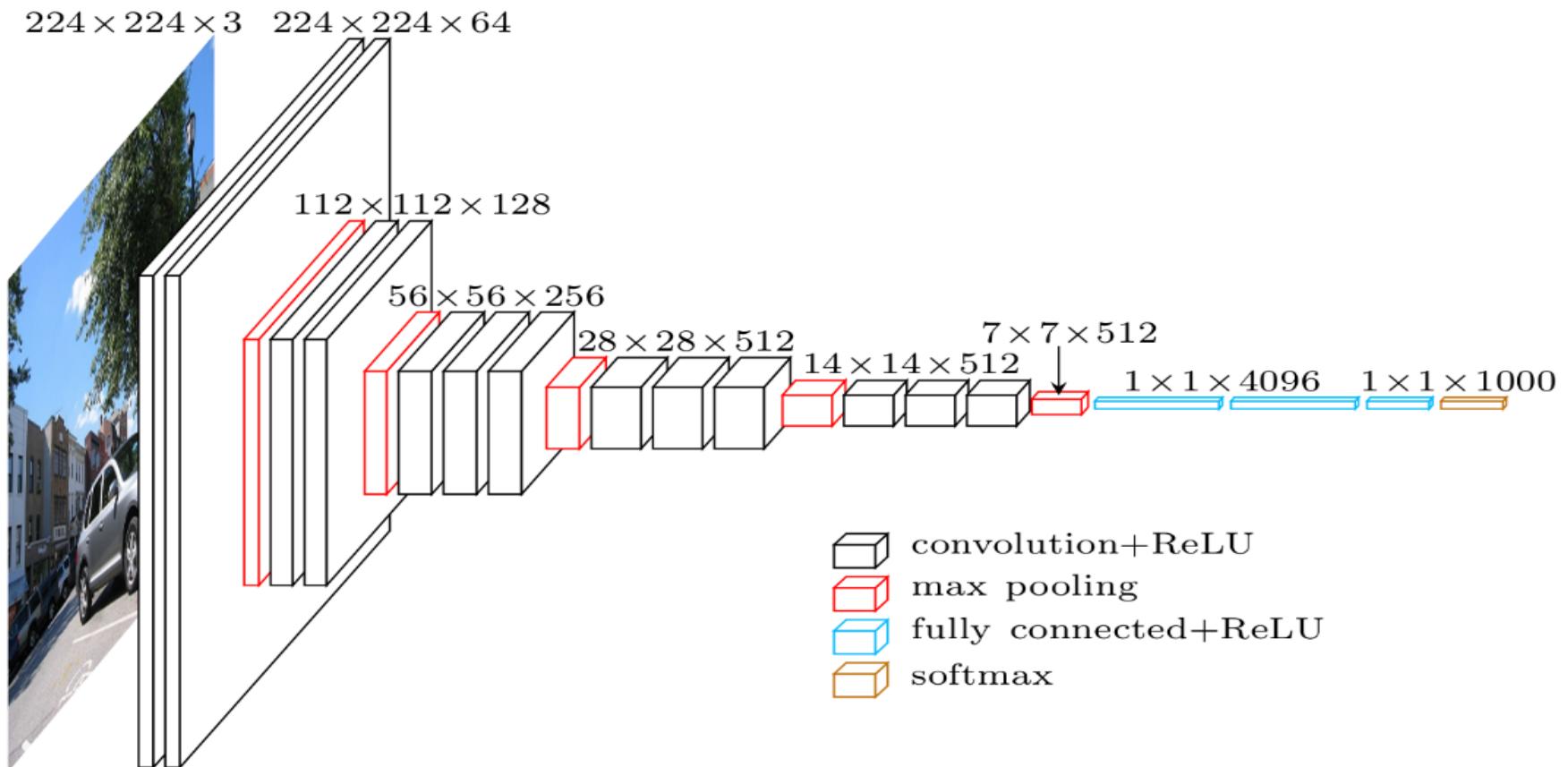
A gagné ImageNet 2012 LSVRC [Krizhevsky, 2012]  
60 million de paramètres appris par SGD  
25 % d'erreurs → 15 % d'erreurs !

4M	<b>FULL CONNECT</b>	4Mflop
16M	<b>FULL 4096/ReLU</b>	16M
37M	<b>FULL 4096/ReLU</b>	37M
	<b>MAX POOLING</b>	
442K	<b>CONV 3x3/ReLU 256fm</b>	74M
1.3M	<b>CONV 3x3ReLU 384fm</b>	224M
884K	<b>CONV 3x3/ReLU 384fm</b>	149M
	<b>MAX POOLING 2x2sub</b>	
	<b>LOCAL CONTRAST NORM</b>	
307K	<b>CONV 11x11/ReLU 256fm</b>	223M
	<b>MAX POOL 2x2sub</b>	
	<b>LOCAL CONTRAST NORM</b>	
35K	<b>CONV 11x11/ReLU 96fm</b>	105M



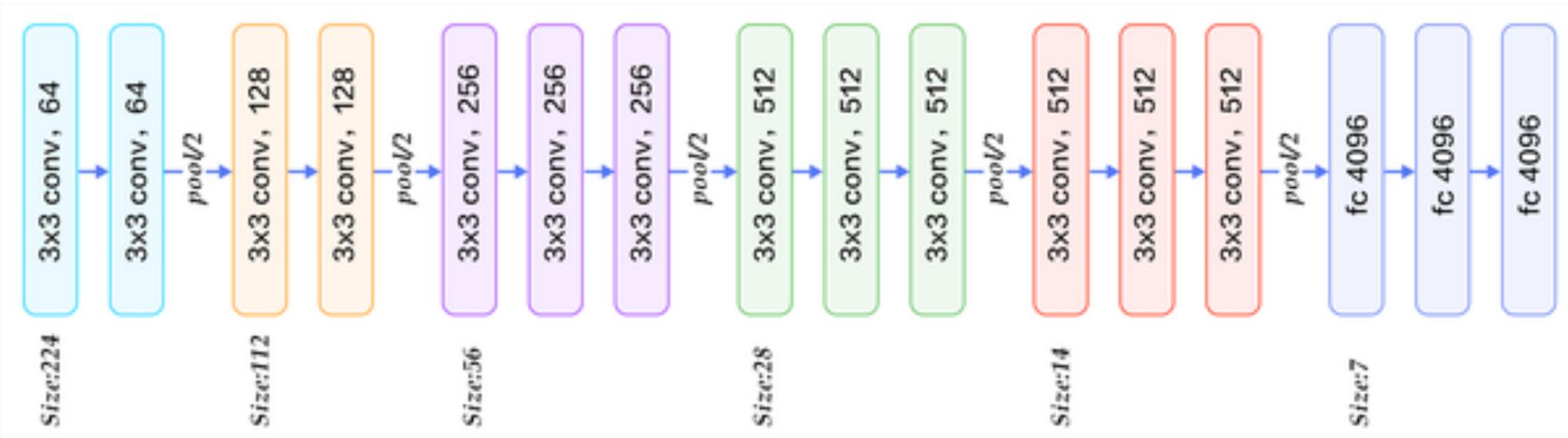
# VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

- [Simonyan & Zisserman, 2015] (Oxford)
- Plusieurs variantes : VGG16, VGG19, ...



# VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

- [Simonyan & Zisserman, 2015] (Oxford)
- Plusieurs variantes : VGG16, VGG19, ...



# ResNet - 2015

- [He et al., 2015] (Microsoft) <https://arxiv.org/abs/1512.03385>

- Residual connexions

- Ajout de connections en parallèle
- Pas/peu de paramètres supplémentaires
- Réduit le gradient vanishing

- Architecture en blocs

- Resnet 18, 32, 50, 164, ..., 1001 !

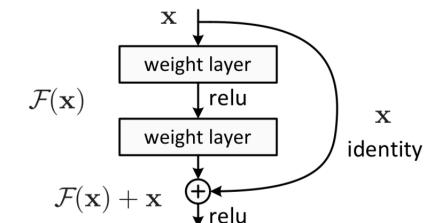
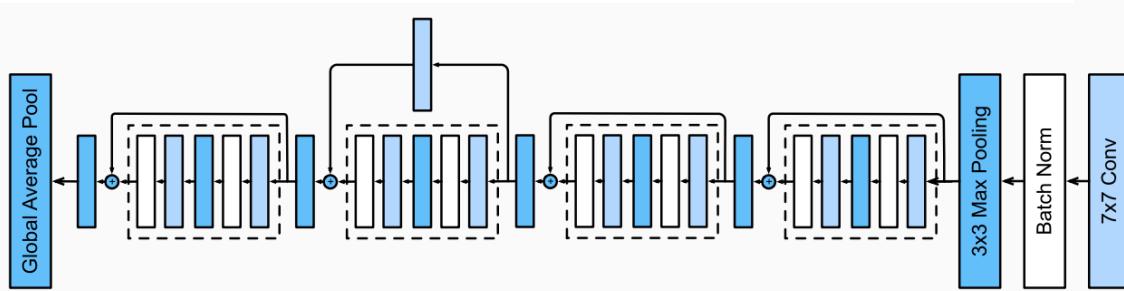
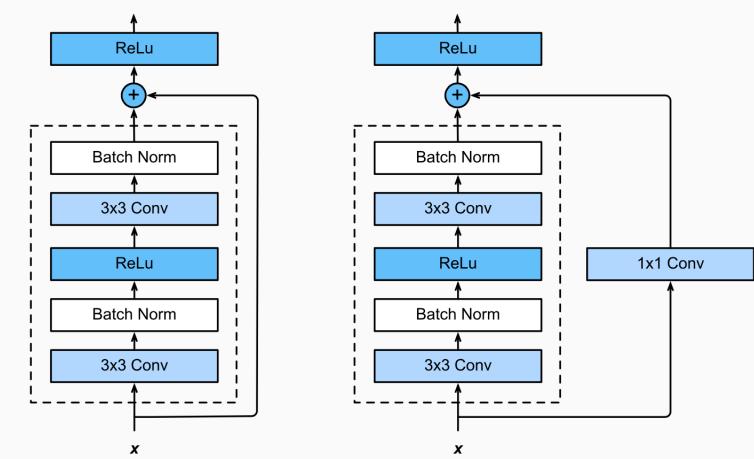
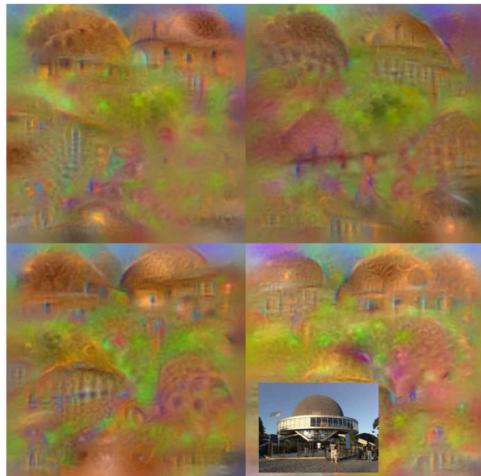


Figure 2. Residual learning: a building block.



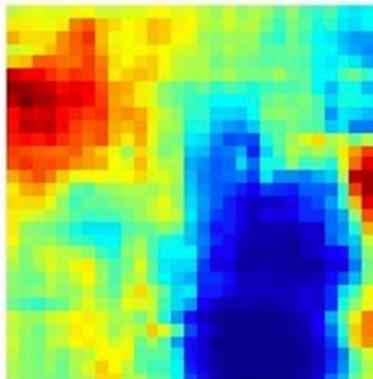
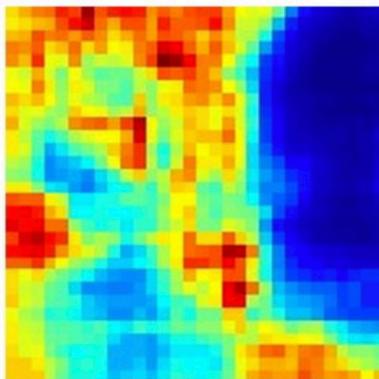
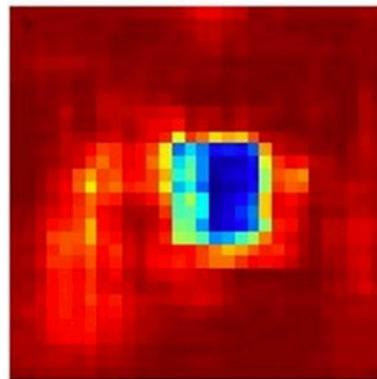
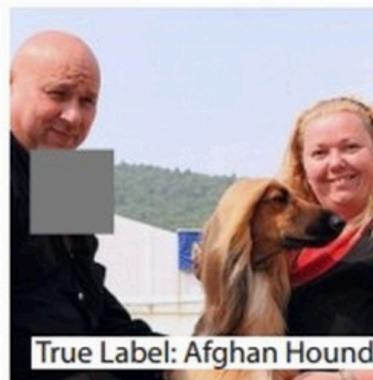
# Visualisation des CNN

- Que fait un filtre ?
  - Trouver / Générer l'image qui maximise l'activation d'un neurone
  - <https://www.robots.ox.ac.uk/~vedaldi/assets/pubs/mahendran16visualizing.pdf>



# Visualisation des CNN

- Visualiser les features map (heat map)
  - Occlusion de parties de l'image pour déterminer les zones qui maximise une classe
  - <https://arxiv.org/pdf/1311.2901.pdf>



# Visualisation des CNN

- Visualiser les features map (heat map)

- Mécanisme d'attention (spatiale)

<https://arxiv.org/abs/1502.03044>



A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



A stop sign is on a road with a mountain in the background.