SORBONNE UNIVERSITÉ

C5. Les réseaux convolutionnels

Advanced Machine Learning (MLA)

Kévin Bailly

kevin.bailly@sorbonne-universite.fr

http://people.isir.upmc.fr/bailly/

Limites des réseaux MLP

- Comment traiter des images de grande dimension ?

```python
class MNISTModel(nn.Module):
    def __init__(self):
        super(MNISTModel, self).__init__()
        self.flatten = nn.Flatten()
        self.fc1 = nn.Linear(28*28, 120)
        self.fc2 = nn.Linear(120, 80)
        self.fc3 = nn.Linear(80, 10)

    def forward(self, x):
        x = self.flatten(x)
        x = torch.sigmoid(self.fc1(x))
        x = torch.sigmoid(self.fc2(x))
        x = self.fc3(x)
        return x
```

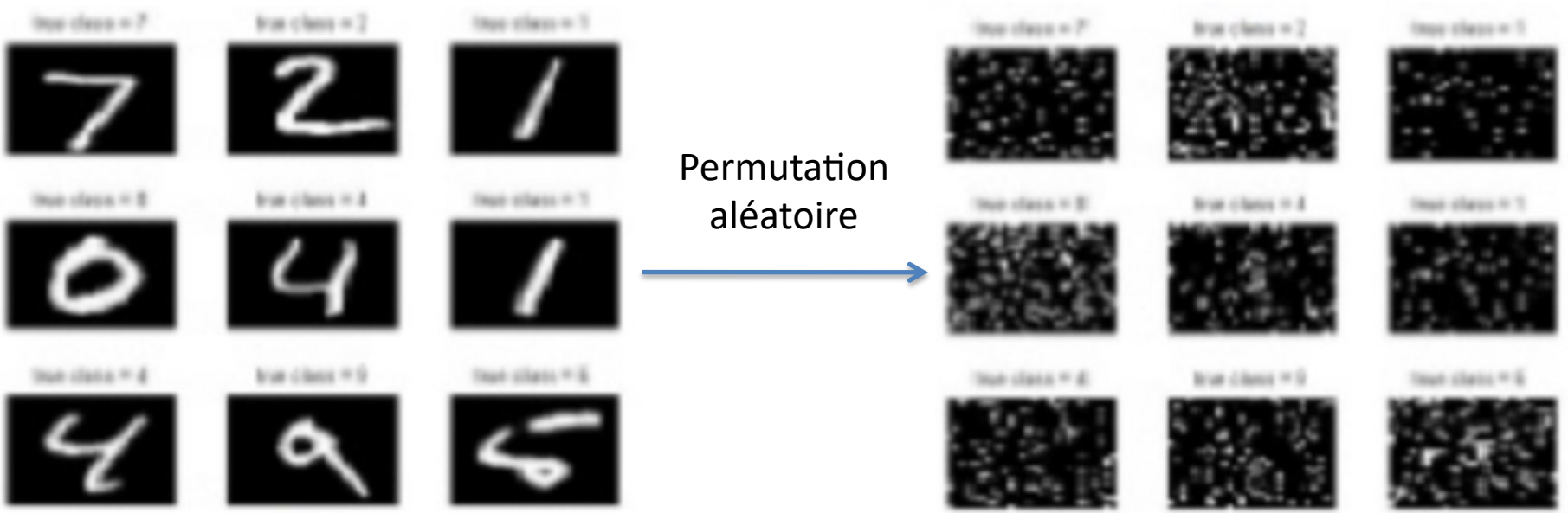
Combien de paramètres pour la première couche cachée ?

28 x 28 x 120 = 94080

Le nombre de paramètres explose pour des images de plus grande résolution en RGB - Ex : 640x480x3x120 = 11M !!

Limites des réseaux MLP

- Comment conserver la structure spatiale ?



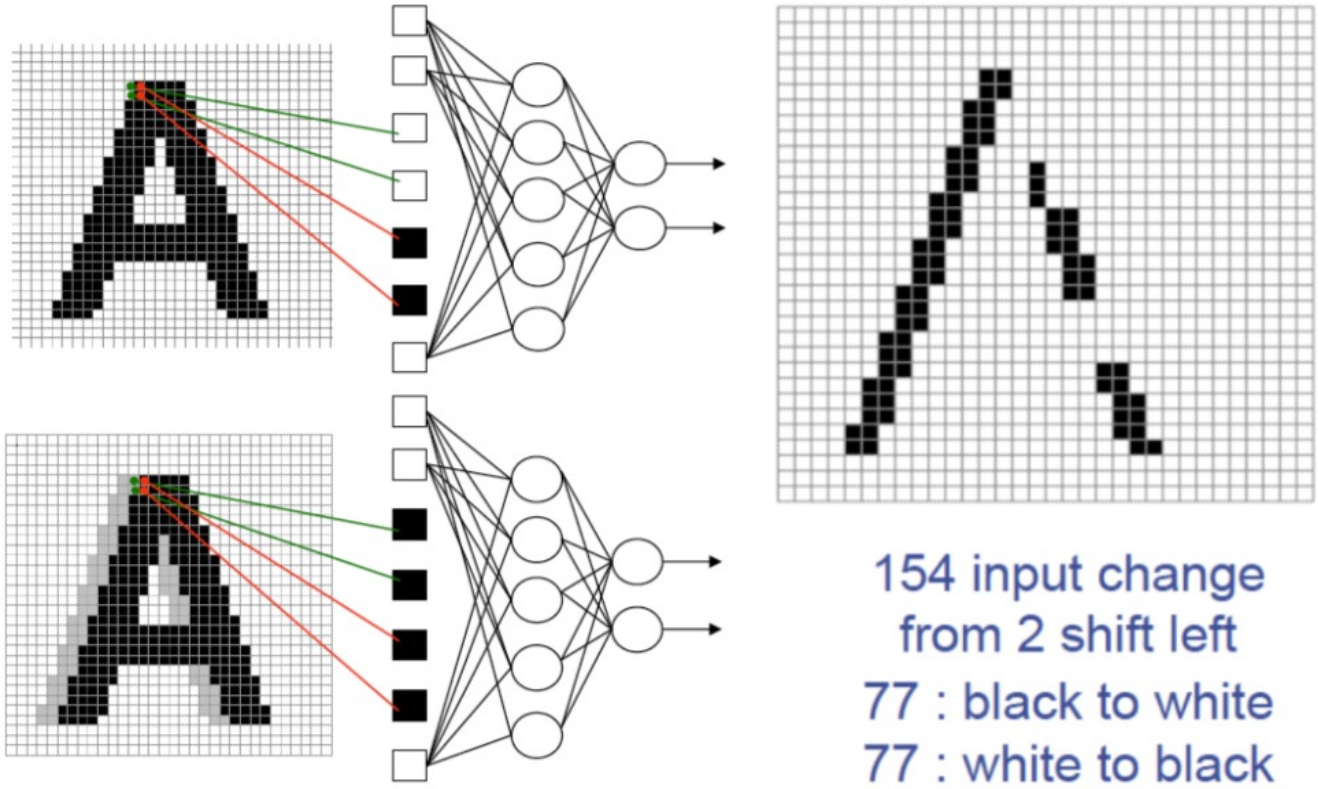
- Un MLP donne-t-il les mêmes résultats sur les 2 bases de données ?

  Le codage vectoriel (*flatten*) ne tient pas compte de la structure 2D

  ➔ la structure locale n'est pas prise en compte !

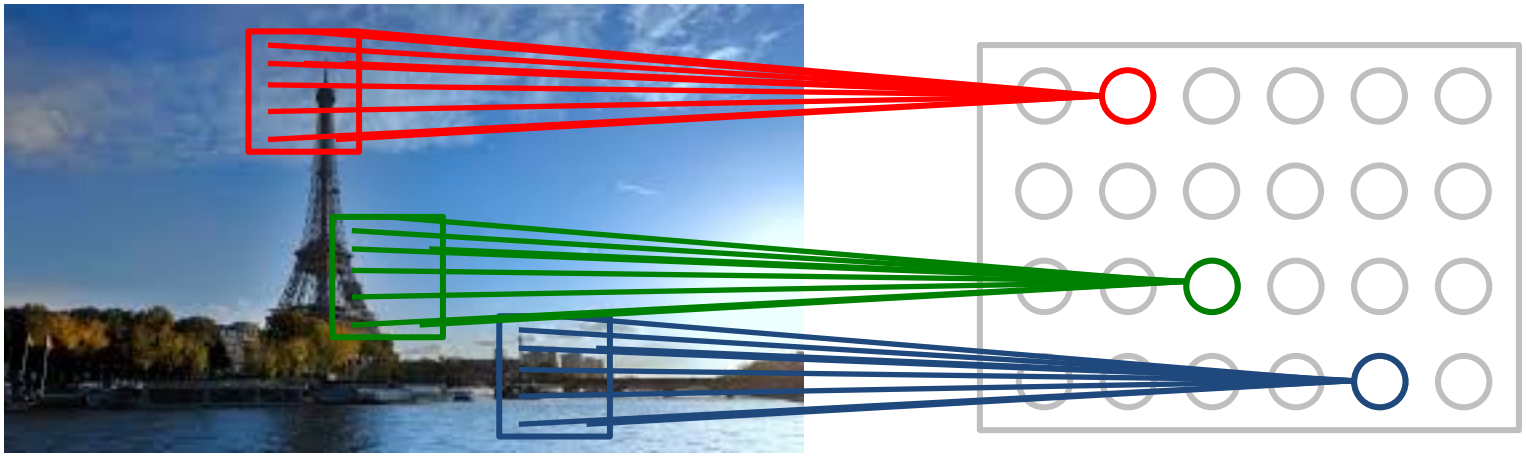
Limites des réseaux MLP

- Comment être moins sensible aux faibles variations



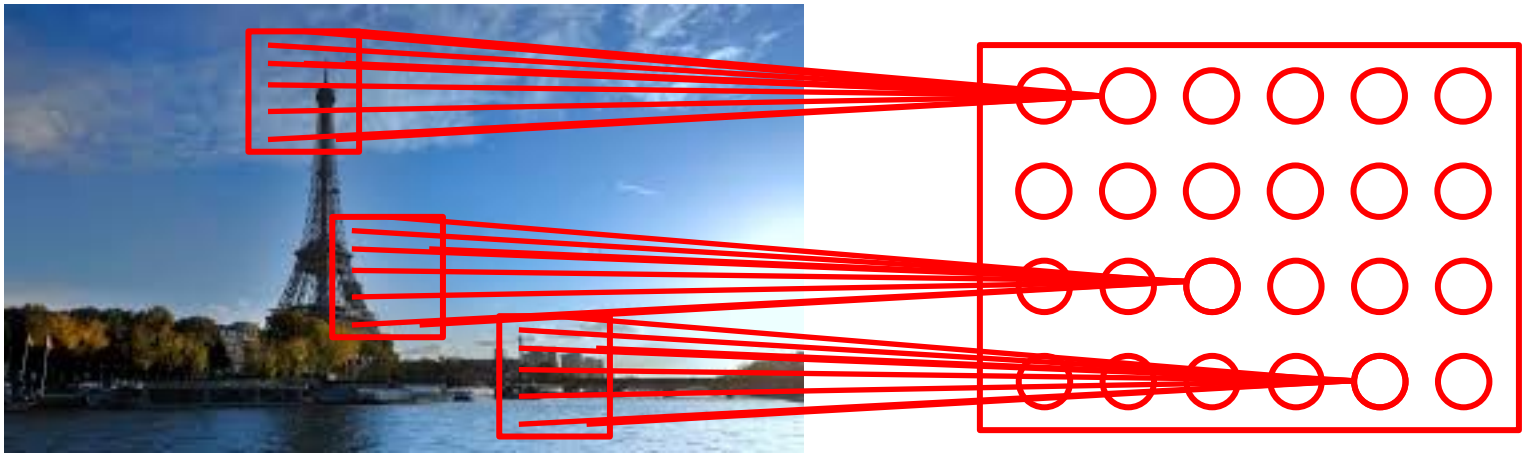
Les réseaux à convolution (CNN)

- Les CNN visent à répondre à ces limitations
  - **Connexions locales** : un neurone est connecté à une sous région (patches locaux)
    - Conservation d'une information spatialement structurée (réponse à des motifs locaux)



Les réseaux à convolution (CNN)

- Les CNN visent à répondre à ces limitations
  - **Connexions locales** : un neurone est connecté à une sous région (patches locaux)
  - **Poids partagés** : même opération répliquée sur tout les neurones
    - Réduction du nombre de paramètres
    - **Convolution avec des filtres appris**



Convolutions

- Chaque neurone (vert foncé) ne « voit » qu'une portion de l'image (bleu foncé)

$$h^n(i,j) = \left(h^{n-1} * w_k\right)(i,j) = \sum_{n=-\frac{d-1}{2}}^{\frac{d-1}{2}} \sum_{m=-\frac{d-1}{2}}^{\frac{d-1}{2}} h^{n-1}(i+n, j+m) * w(n,m)$$

Corrélation croisée

Poids du réseau w (filtre)

0	1	2
2	2	0
0	1	2

$*$

Image en entrée

3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

$=$

Résultat de la convolution

12.0	12.0	17.0
10.0	17.0	19.0
9.0	6.0	14.0

- On applique une non-linéarité (ex. ReLU) sur tous les pixels de la carte d'activation

Convolutions

- Chaque neurone (vert foncé) ne « voit » qu'une portion de l'image (bleu foncé)

$$h^n(i,j) = \left(h^{n-1} * w_k\right)(i,j) = \sum_{n=-\frac{d-1}{2}}^{\frac{d-1}{2}} \sum_{m=-\frac{d-1}{2}}^{\frac{d-1}{2}} h^{n-1}(i+n, j+m) * w(n,m)$$

Corrélation croisée

Poids du réseau w (filtre)

0	1	2
2	2	0
0	1	2

$*$

Image en entrée

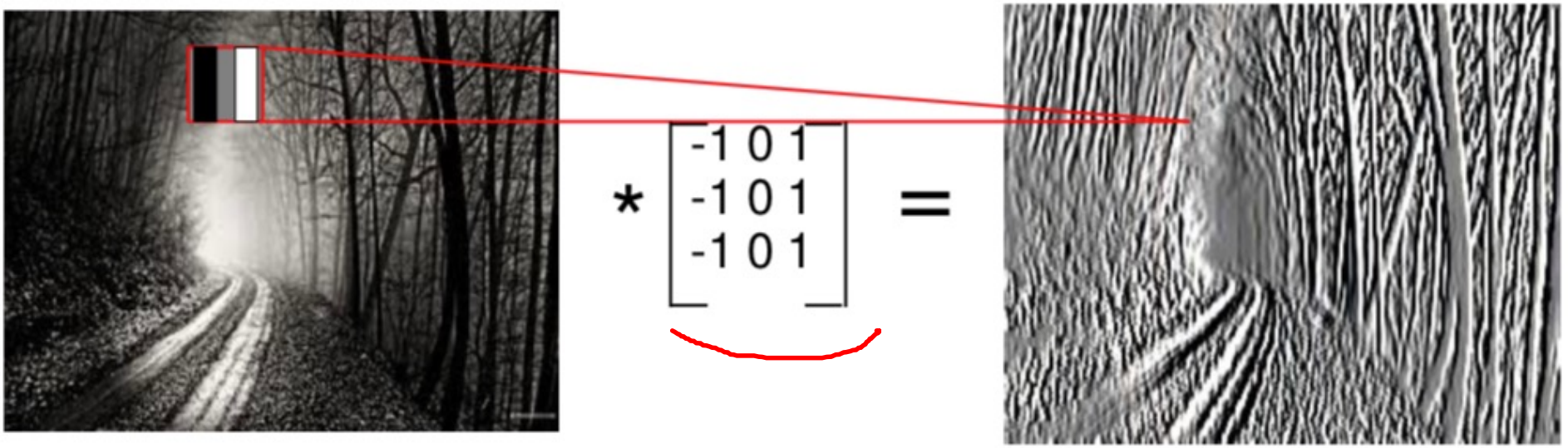
3_0	3_1	2_2	1	0
0_2	0_2	1_0	3	1
3_0	1_1	2_2	2	3
2	0	0	2	2
2	0	0	0	1

$=$

Résultat de la convolution

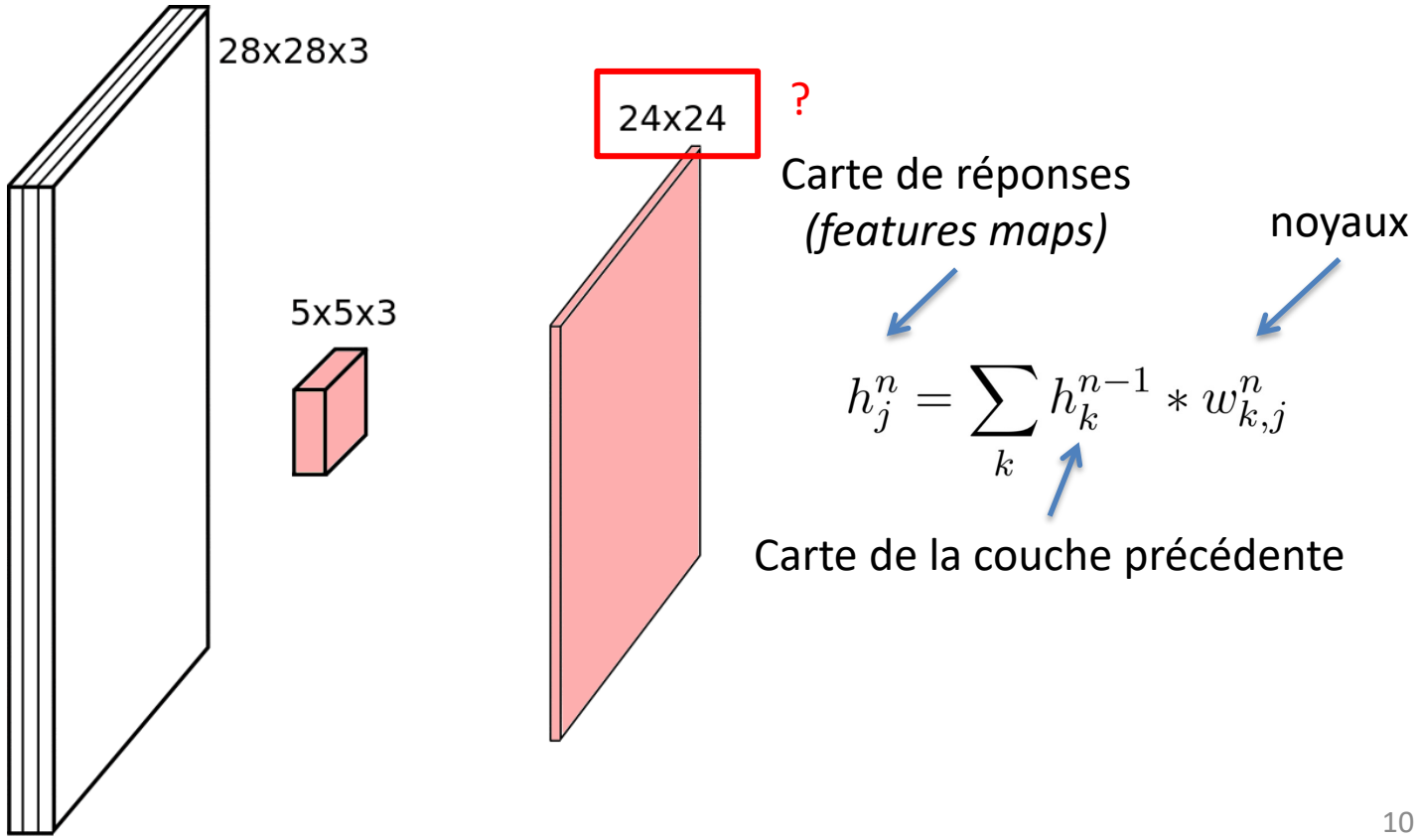
12.0	12.0	17.0
10.0	17.0	19.0
9.0	6.0	14.0

Convolutions



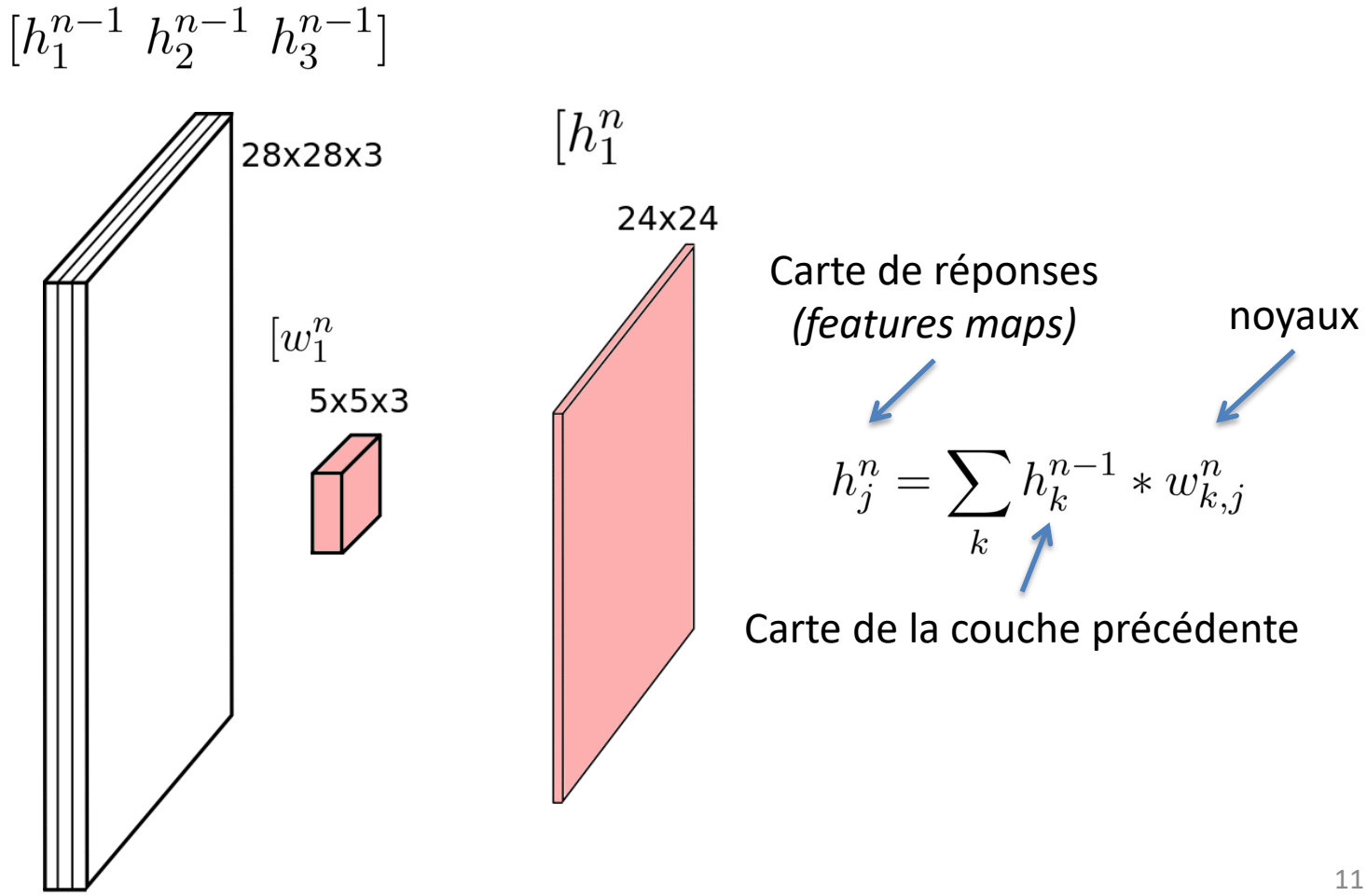
Couches de convolutions

- Une image est un tenseur de taille [h*w*c]
- Les convolutions sont calculées pour chaque canal c puis sommées



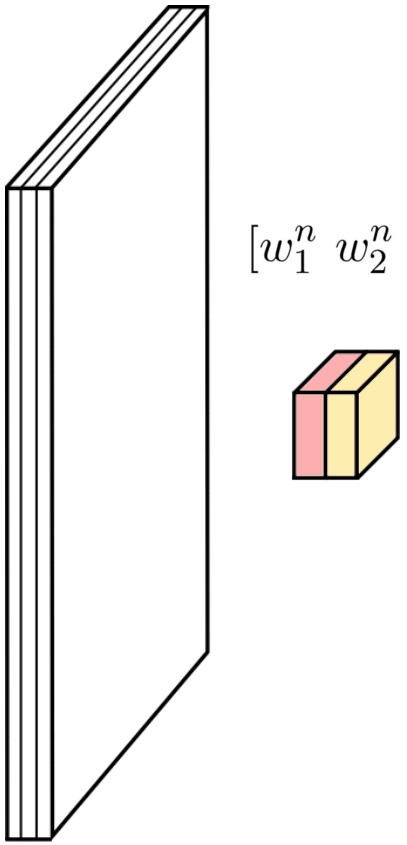
[Ollion et al. 2018]

Couches de convolutions



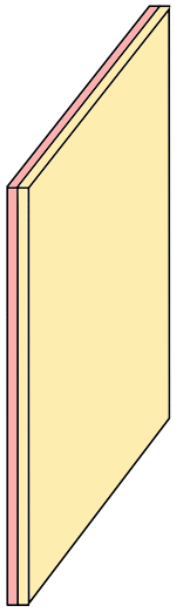
Couches de convolutions

$[h_1^{n-1}\ h_2^{n-1}\ h_3^{n-1}]$



$[w_1^n\ w_2^n$

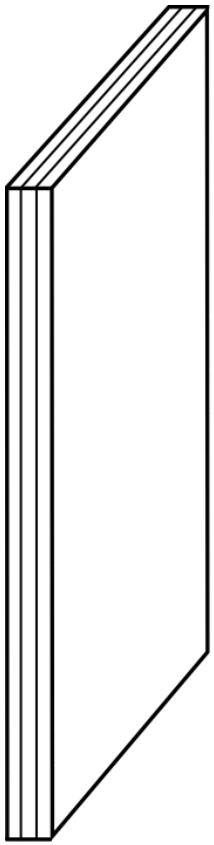
$[h_1^n\ h_2^n$



$$h_j^n = \sum_k h_k^{n-1} * w_{k,j}^n$$

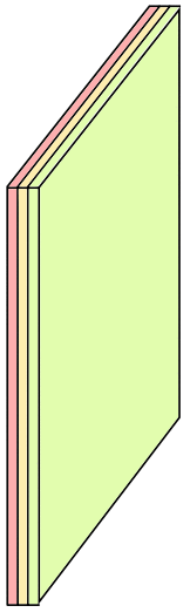
Couches de convolutions

$[h_1^{n-1}\ h_2^{n-1}\ h_3^{n-1}]$



$[w_1^n\ w_2^n\ w_3^n$

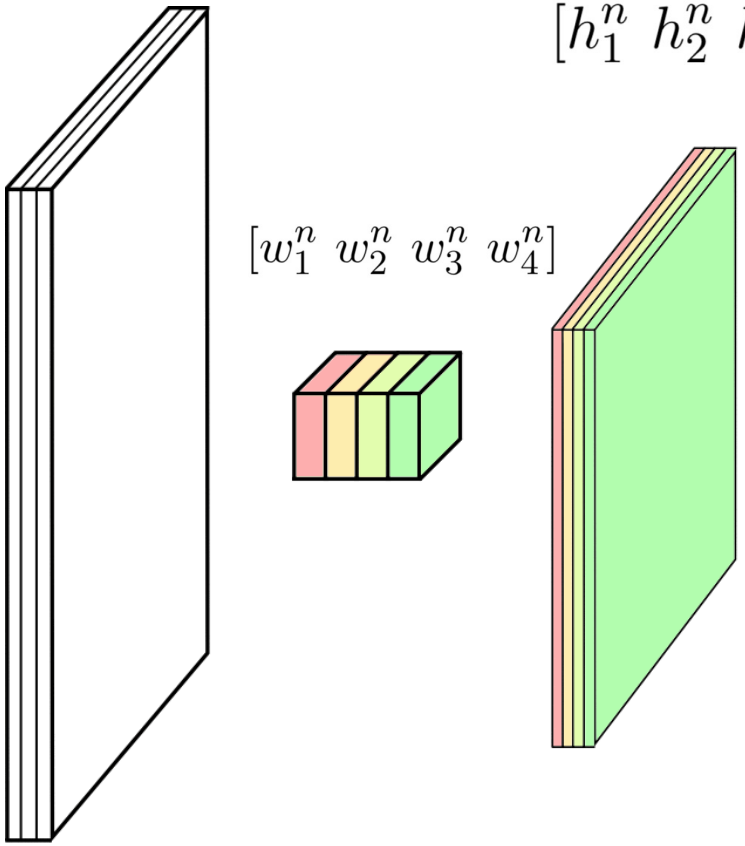
$[h_1^n\ h_2^n\ h_3^n$



$$h_j^n = \sum_k h_k^{n-1} * w_{k,j}^n$$

Couches de convolutions

$[h_1^{n-1}\ h_2^{n-1}\ h_3^{n-1}]$



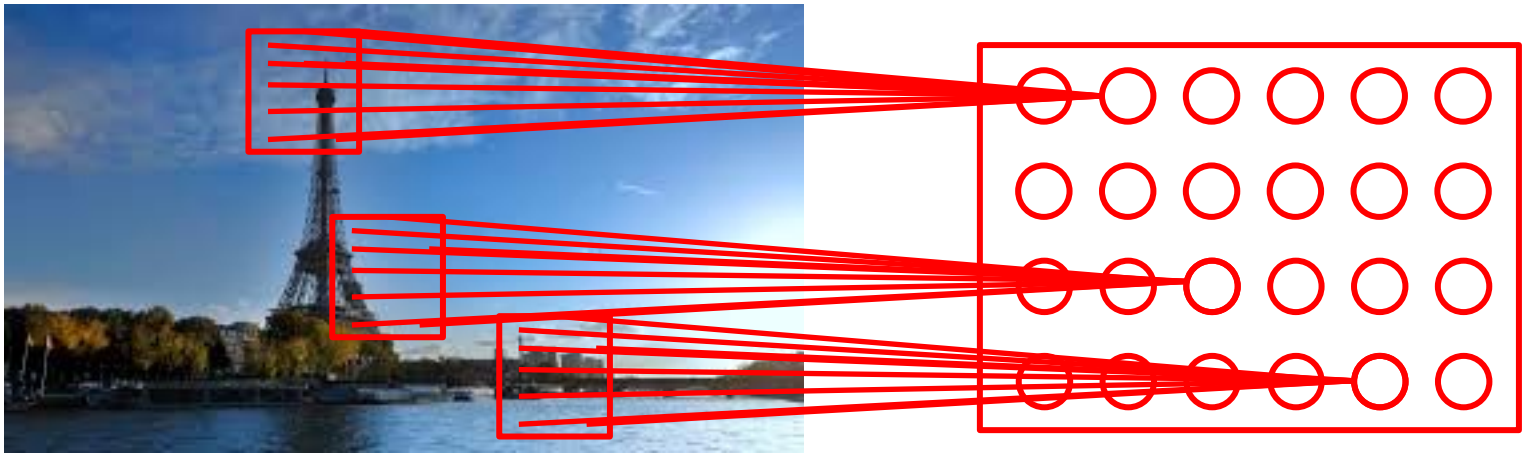
$[h_1^n\ h_2^n\ h_3^n\ h_4^n]$

$[w_1^n\ w_2^n\ w_3^n\ w_4^n]$

$$h_j^n = \sum_k h_k^{n-1} * w_{k,j}^n$$

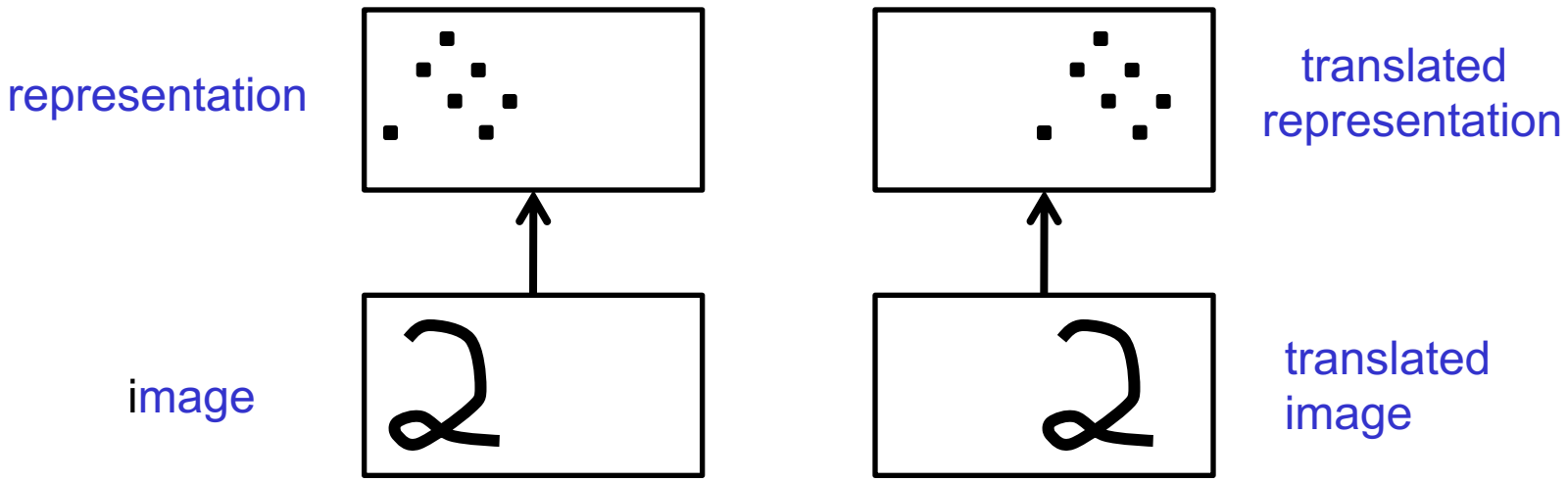
Les réseaux à convolution (CNN)

- Les CNN visent à répondre à ces limitations
  - **Connexions locales** : un neurone est connecté à une sous région (patches locaux)
  - **Poids partagés** : même opération répliquée sur tout les neurones
  - **Et l'invariance aux faibles variations ??**



Equivariance

- Les poids partagés (noyaux) n'assurent **pas une invariance** à la translation mais **une equivariance**

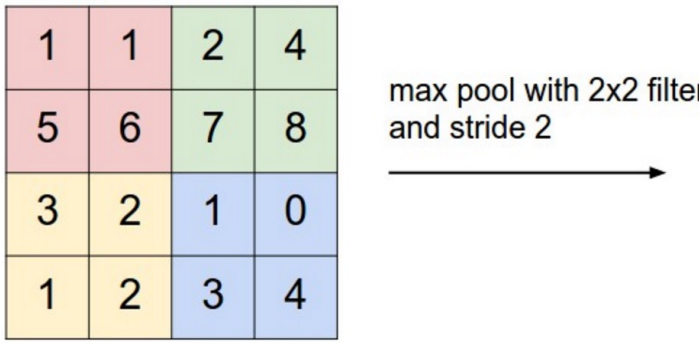


- Comment introduire une invariance aux transformations spatiales 2D ??

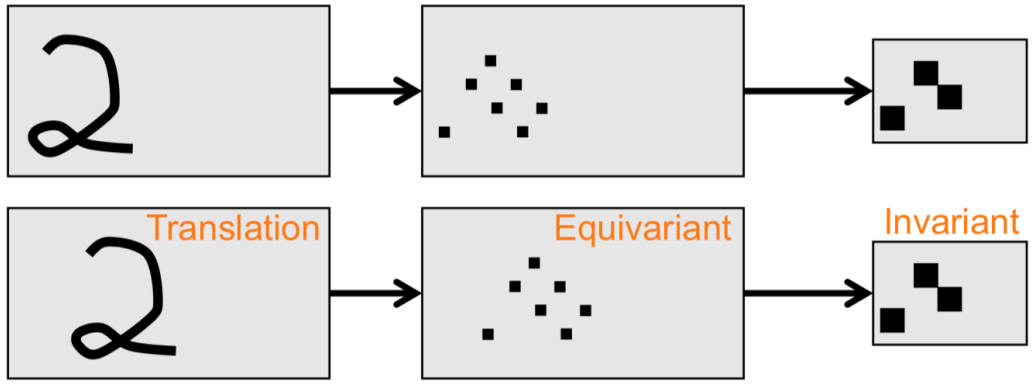
[Hinton]

L'étape de *Pooling*

- Un **pooling spatial** (local) des cartes de réponses



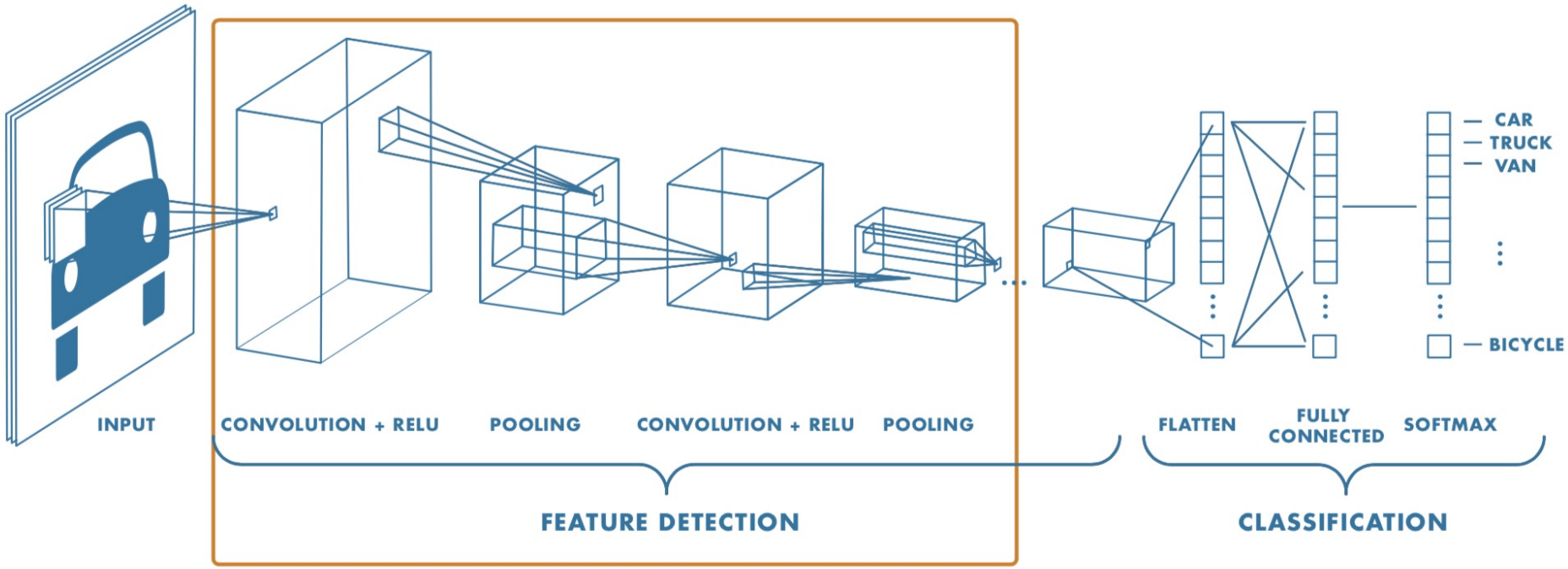
[F.F. Li, cs231]



- Réduit la dimension des cartes d'activation
- Assure une invariance au faibles translations
- Il existe plusieurs types de pooling (maxpool, meanpool, random…)

Réseau CNN

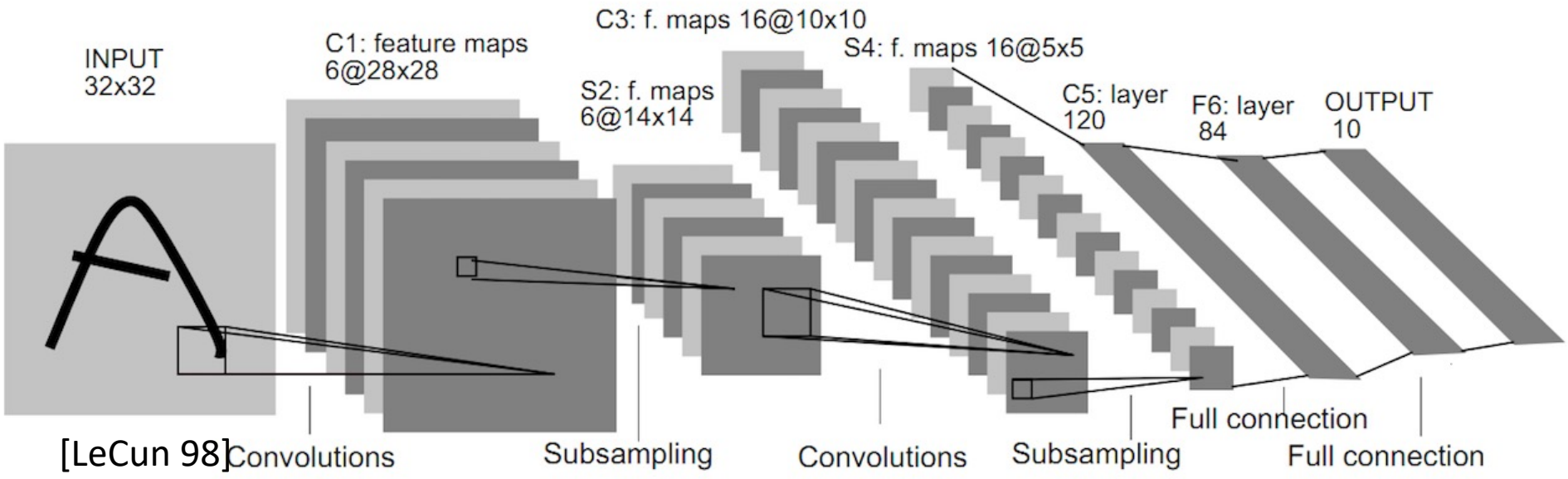
- Vue globale d'un réseau CNN pour des problèmes de classification



[Mathworks : Introducing Deep Learning with MATLAB]

LeNet-5

- Architecture :
  - 3 couches de convolution C1, C3 et C5 : taille des filtres ?
  - 2 couches de « average pooling » : taille ?
  - 1 couche complètement connectée (FC, *fully connceted*) F6
  - Fonction d'activation : tanh



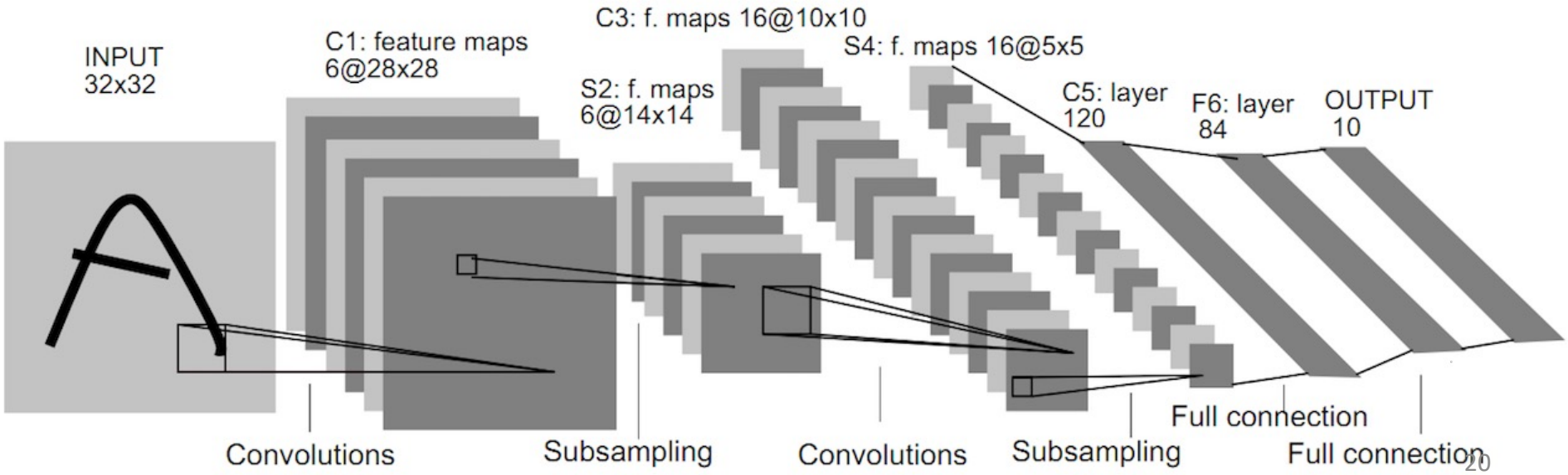
[LeCun 98]

LeNet-5 : spécificités

- 1 couche de sortie « atypique » : 10 fonctions gaussiennes (RBF)

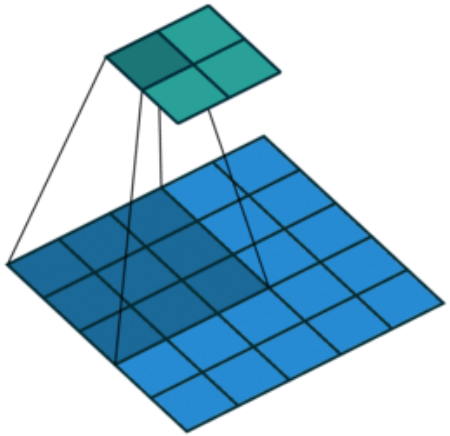
- Couches S2 -> C3 partiellement connectées
  → Réduction du nombre de paramètres
  → Décorrélation des cartes

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

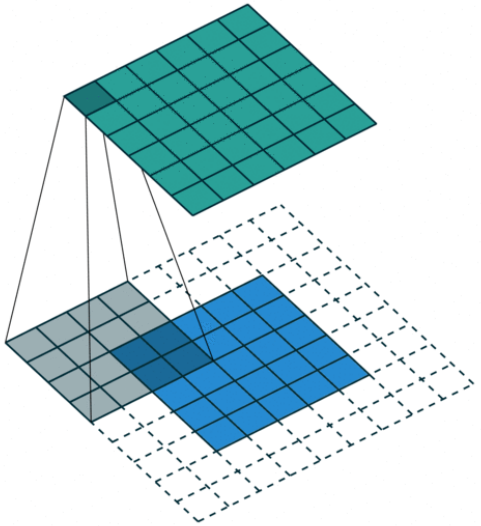


Stride, padding et dilation

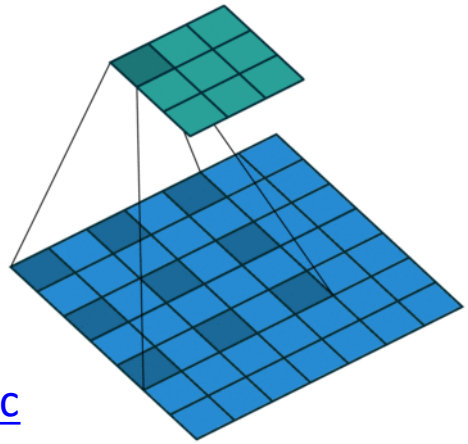
- **Stride**
  - Augmente le pas de l'opérateur de convolution
  - **Effet ?**
  - Réduction de la taille de la carte d'activation



- **Padding**
  - Ajoute un bord à l'image (généralement 0)
  - Conserve la taille de l'image d'origine



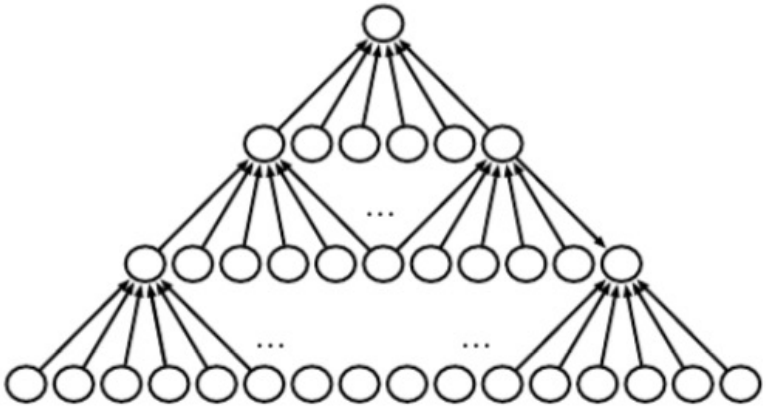
- **Dilation**
  - Réduit la taille
  - ~ (conv + pool)



https://github.com/vdumoulin/conv_arithmetic

Effet du padding

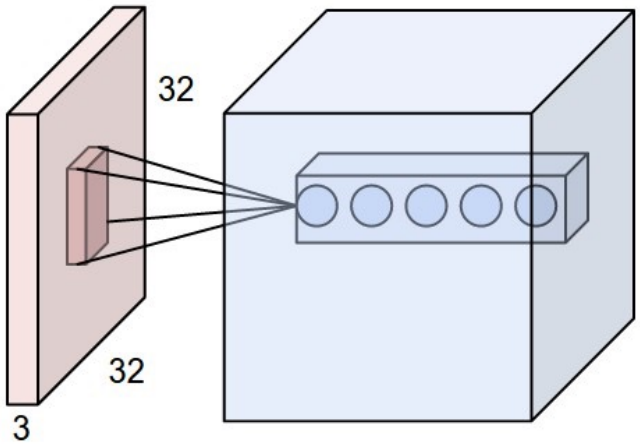
Sans

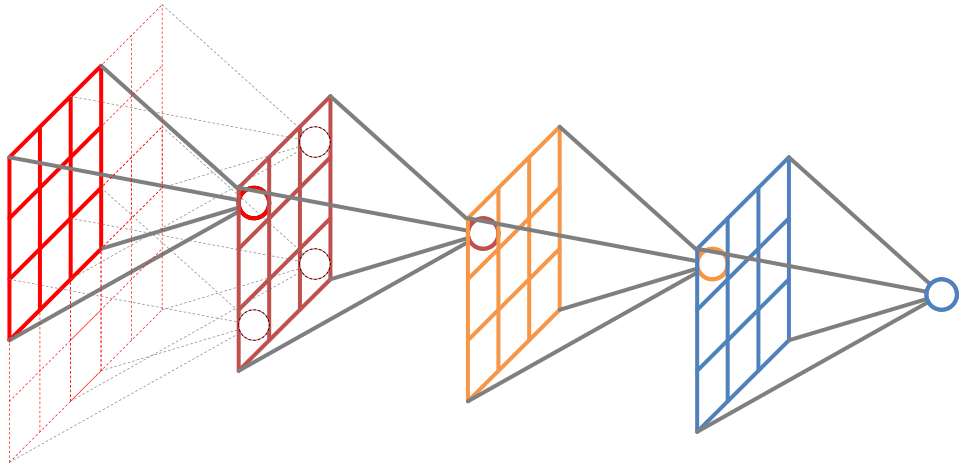


Avec

Champs réceptifs

- Portion de l'image qui a un impact sur la décision d'un seul neurone

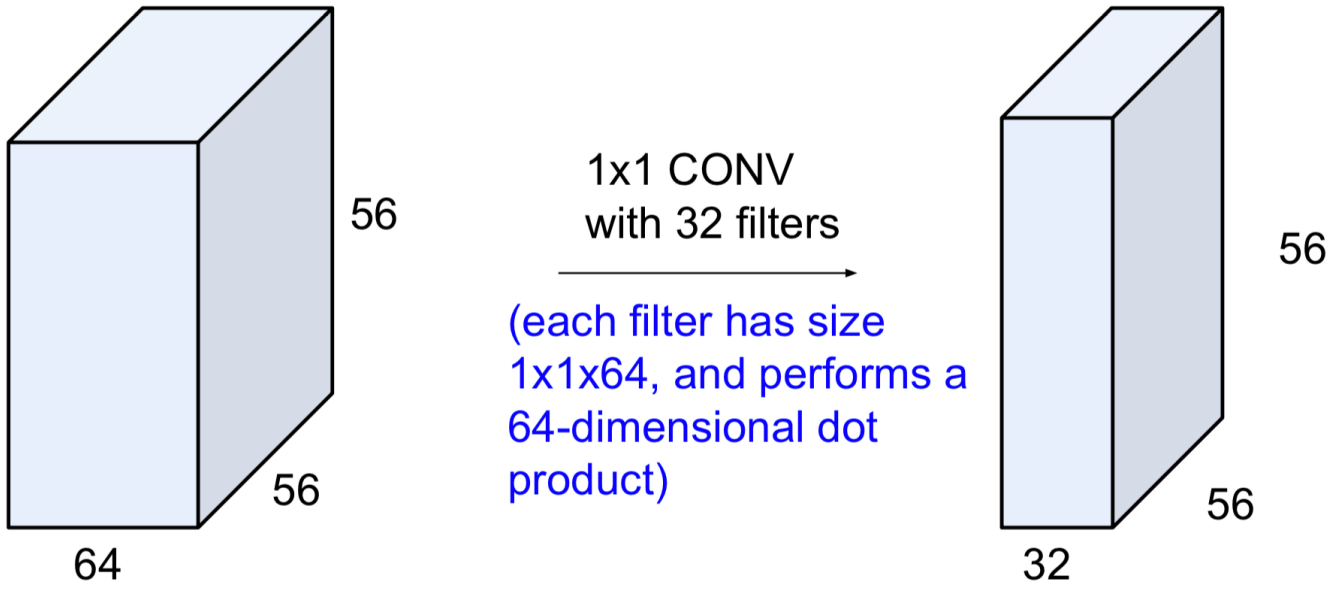




- Comment évolue le champs réceptif avec la profondeur du réseau ?
  - Il augmente (même sans pooling)
- Est-il préférable d'augmenter la profondeur du réseau ou la largeur des filtres ? Pourquoi ?
  - Augmenter la profondeur limite le nombre de paramètres

Champs réceptifs

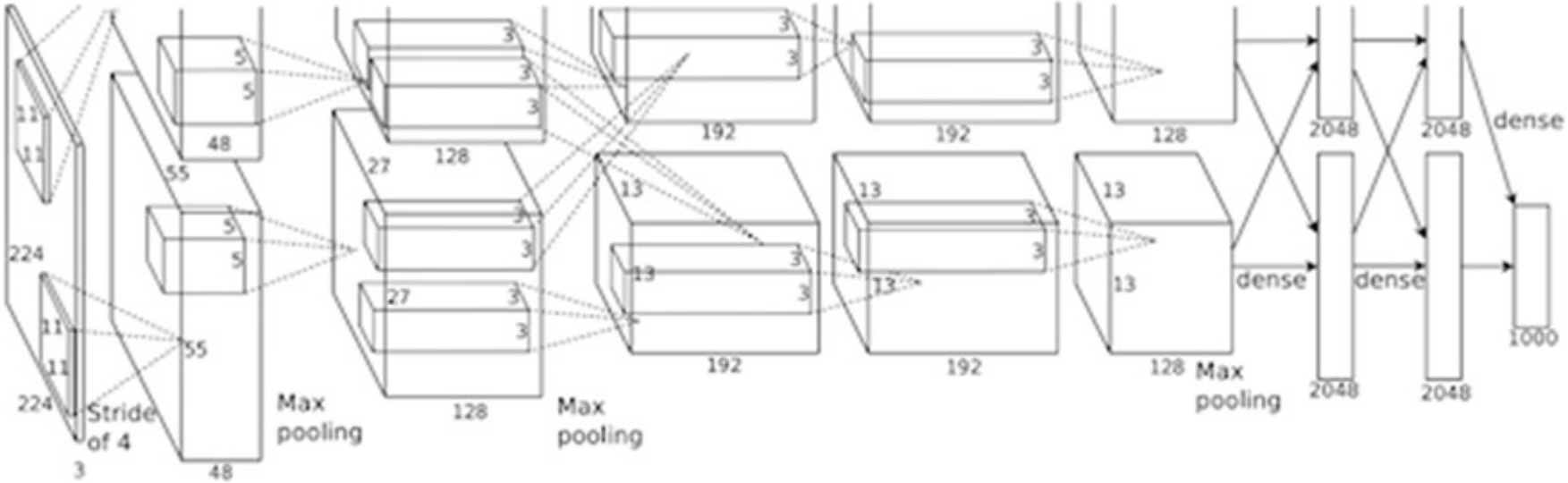
- Certains réseaux utilisent des convolutions 1x1 pourquoi ??



- Permet de compresser l'information entre les canaux
- Pour les réseaux très profonds, séparer les convolutions spatiales et inter-canaux permet de limiter le nombre de paramètres (ex: GoogLeNet)

Quelques architectures modernes

- **AlexNet (2012)**
  - Premier réseaux CNN à remporter LSVRC (Challenge ImageNet)
    - Top 5 test error rate : 15.4%
  - 7 couches
  - ReLU (plus rapide à entrainer) + Dropout (meilleure généralisation)
  - Augmentation de données (translations réflexions-, patches)
  - Gradient stochastique + momentum + weight decay
  - 2 architectures en parallèle pour la répartion sur GPU



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Quelques architectures modernes

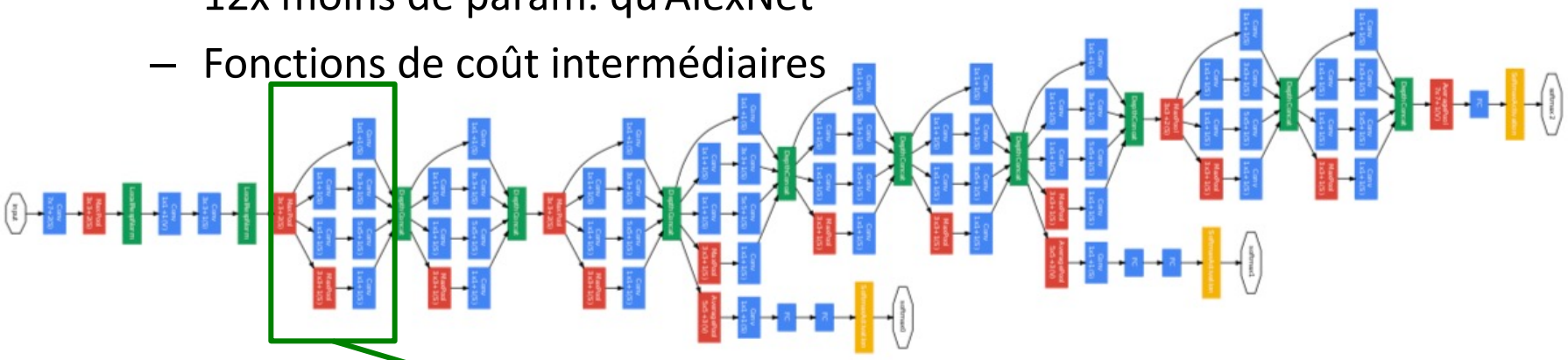
- **VGG Net 2014**
  - % Erreur sur ImageNet : 7.3
  - 16 couches
  - Mais des filtres plus petits (3x3) au lieu de (11x11) et (5x5) pour les 1ère c. AlexNet
    - 3 (3x3) ~ 1(7,7)
  - Deep and simple !!

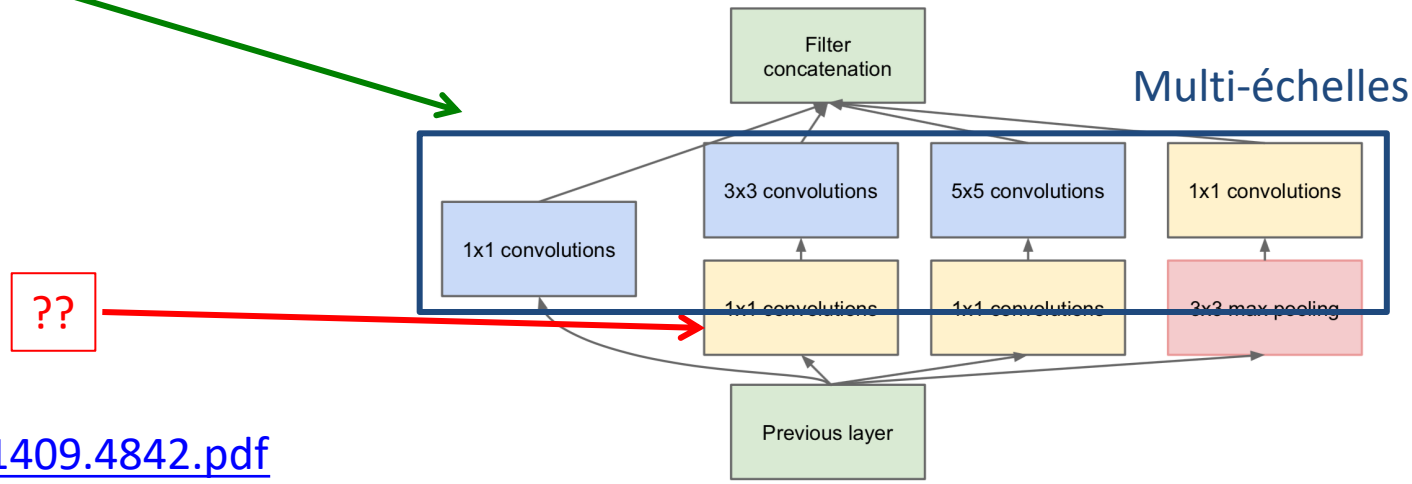
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

https://arxiv.org/pdf/1409.1556v6.pdf

Quelques architectures modernes

- **GoogLeNet 2015**
  - 100 couches, pas de FC (average pooling à la place)
  - 12x moins de param. qu'AlexNet
  - Fonctions de coût intermédiaires

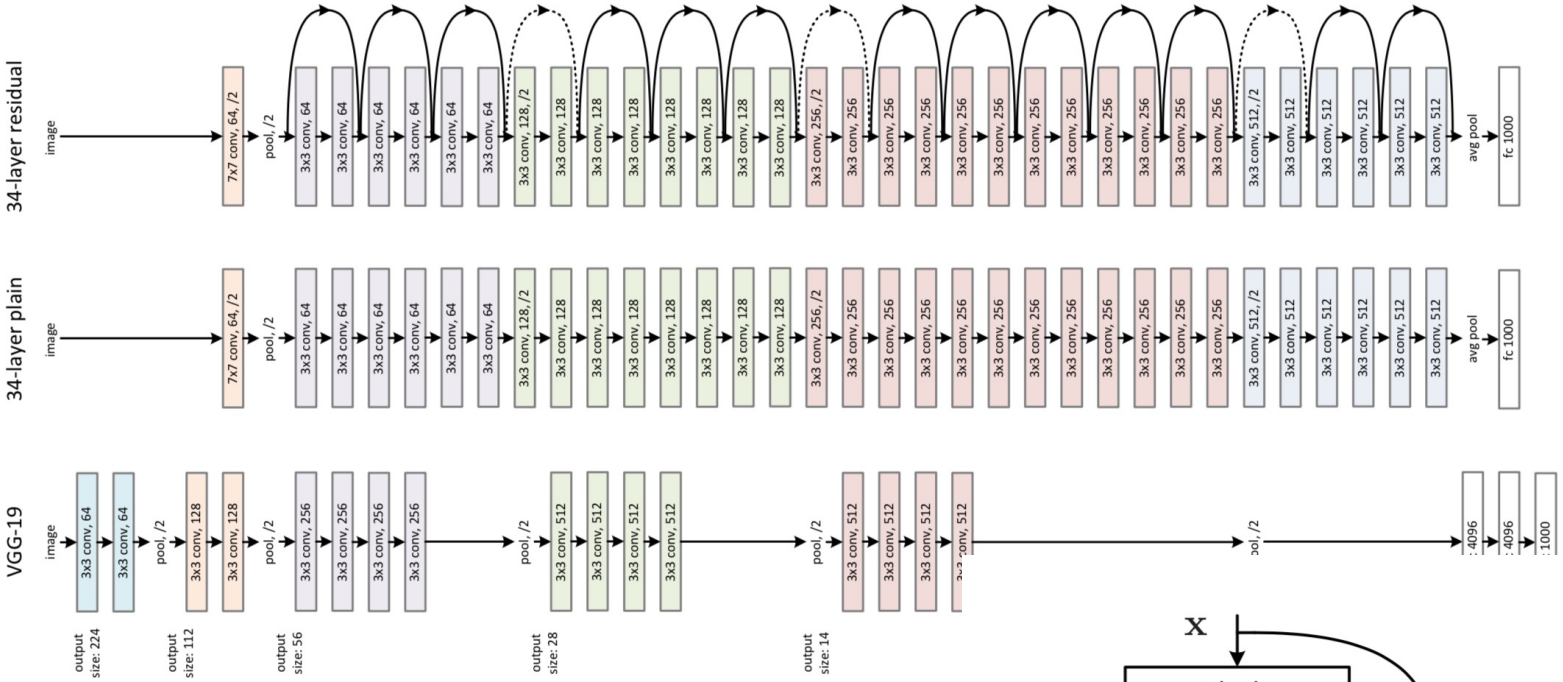


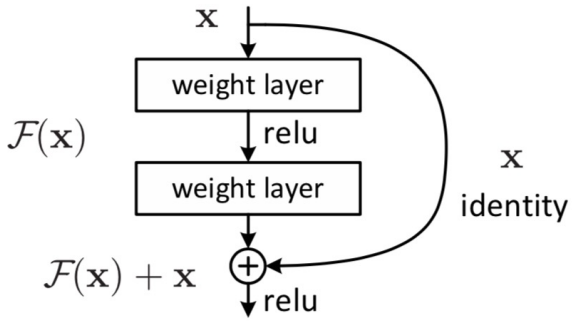


https://arxiv.org/pdf/1409.4842.pdf

Quelques architectures modernes

- Microsoft ResNet (2015)
  - 152 couches avec





Popular CNN Architectures:
https://niessner.github.io/I2DL/slides/10.CNN-2.pdf