

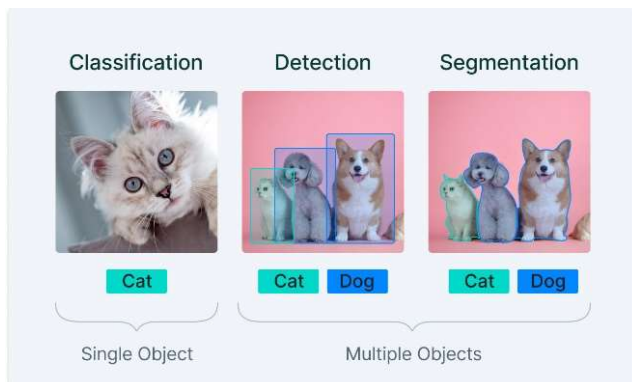
APPRENTISSAGE PROFOND ET IMAGES

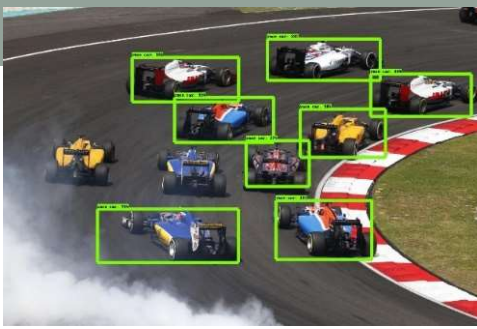
CONVOLUTION NN

Alexandre Meyer¹

¹Equipe SAARA, laboratoire LIRIS

Master ID3D et IA





DEEP LEARNING A PERMIS DES AVANCÉES FORTES EN VISION PAR ORDINATEUR

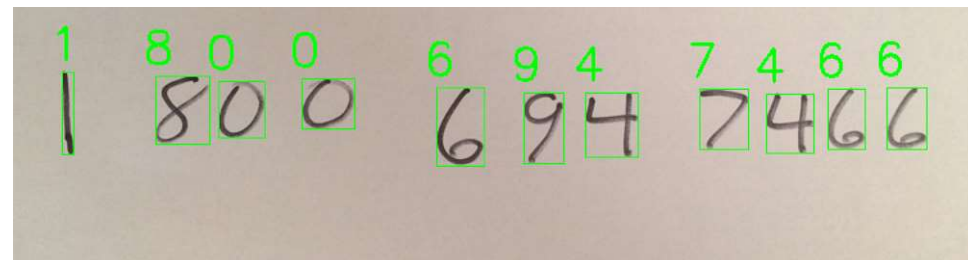
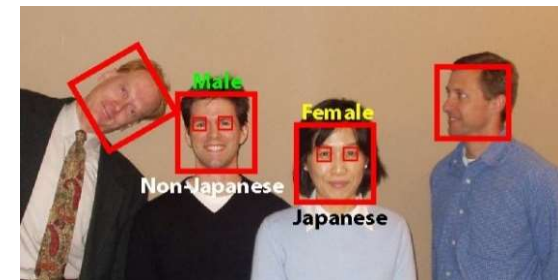
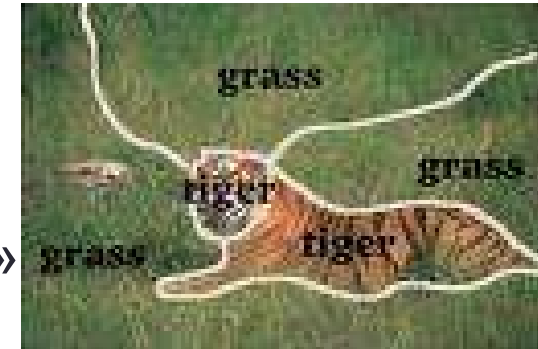
- Classification d'images
- Segmentation
- Détection et suivi d'objets
- Génération d'images
- Reconstruction de pose
- Reconnaissance d'action
- ...



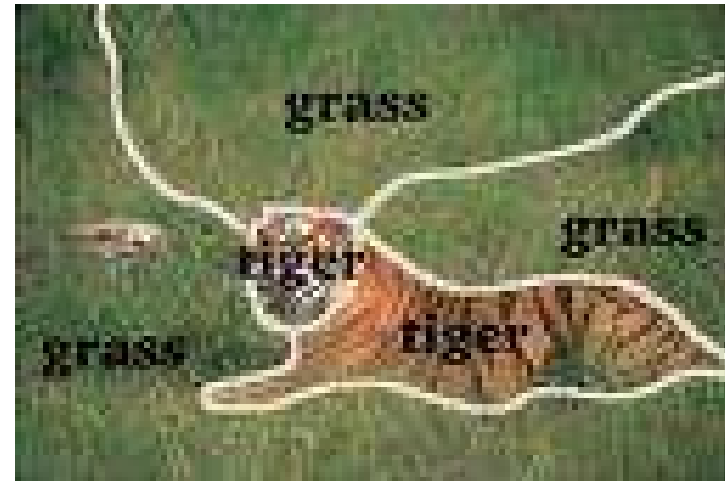
Deep Learning et Reconnaissance

Reconnaissance

- d'une famille d'objets : « c'est un chat »
- d'un objet précis : « c'est Paul », « c'est un 8 »
- d'une expression/style : « le visage sourit »
- ...



Vision de haut niveau : reconnaissance



Qu'est-ce que vous voyez dans cette image ?

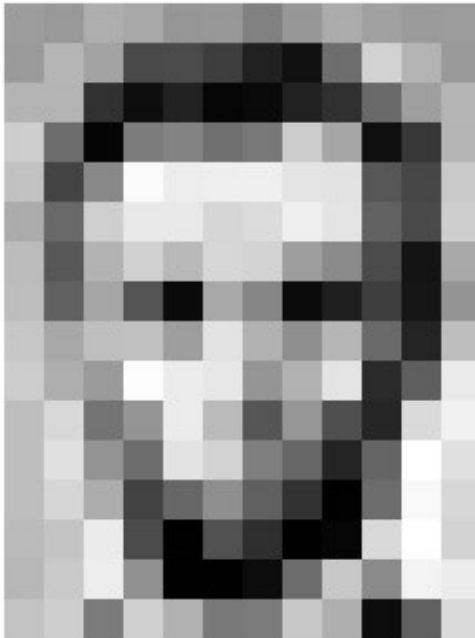
Tâche extrêmement difficile :

le tigre doit être reconnu sous tous les angles, parfois caché, avec des éclairages différents sur chaque photo.

➔ Assez proche du test de Turing

Une image

- Une image est une matrice ou un tab 2D



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	5	10	33	48	105	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	67	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	105	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	95	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	95	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

F



1	1	1
1	1	1
1	1	1

- $$F[x, y]$$

[illegible]

$$G[x, y]$$

A 10x10 grid with a red square in the top-left corner. The red square is located in the first column and the first row, with a side length of 1 unit.

Filtre Gaussien : Blurring

F

$1/9$

1	1	1
1	1	1
1	1	1



Filtre = convolution

- Identité



Original

•0	•0	•0
•0	•1	•0
•0	•0	•0

= ?

- Décalage à droite




Original

•0	•0	•0
•0	•0	•1
•0	•0	•0

= ?

Filtre = convolution

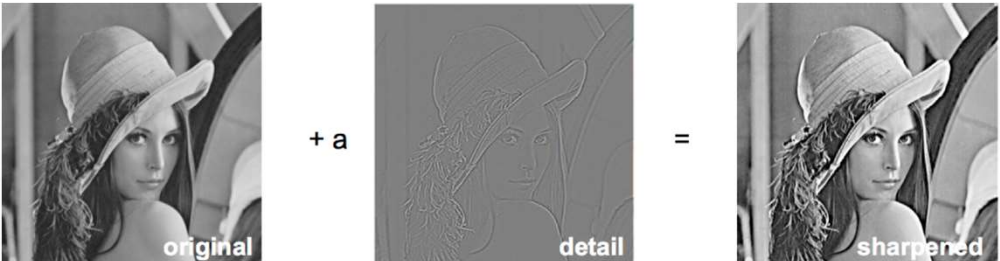
- Original – Smoothed
=> détails



original – smoothed (5x5) = detail

$$\begin{bmatrix} \bullet 0 & \bullet 0 & \bullet 0 \\ \bullet 0 & \bullet 1 & \bullet 0 \\ \bullet 0 & \bullet 0 & \bullet 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \end{bmatrix}$$

- Original + Détails
=> Sharpened



original + a detail = sharpened

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

Filtres / convolution

DEMO

<https://setosa.io/ev/image-kernels/>

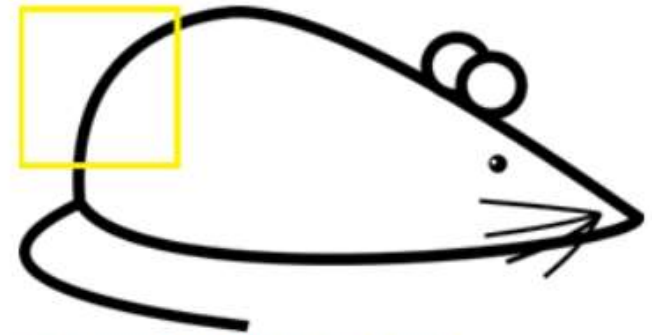
<https://generic-github-user.github.io/Image-Convolution-Playground/src/>

Filtres

- Détecteur de courbes



Original image



Visualization of the filter on the image



Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

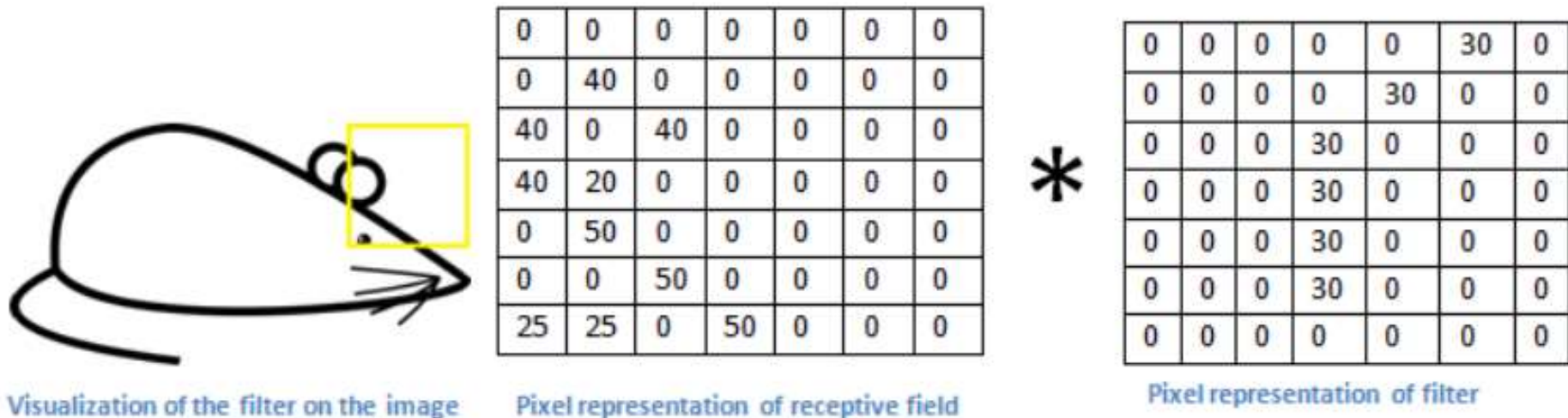
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = $(50 \times 30) + (50 \times 30) + (50 \times 30) + (20 \times 30) + (50 \times 30) = 6600$ (A large number!)

Filtre : détecteur de courbes

- Le même filtre ailleurs

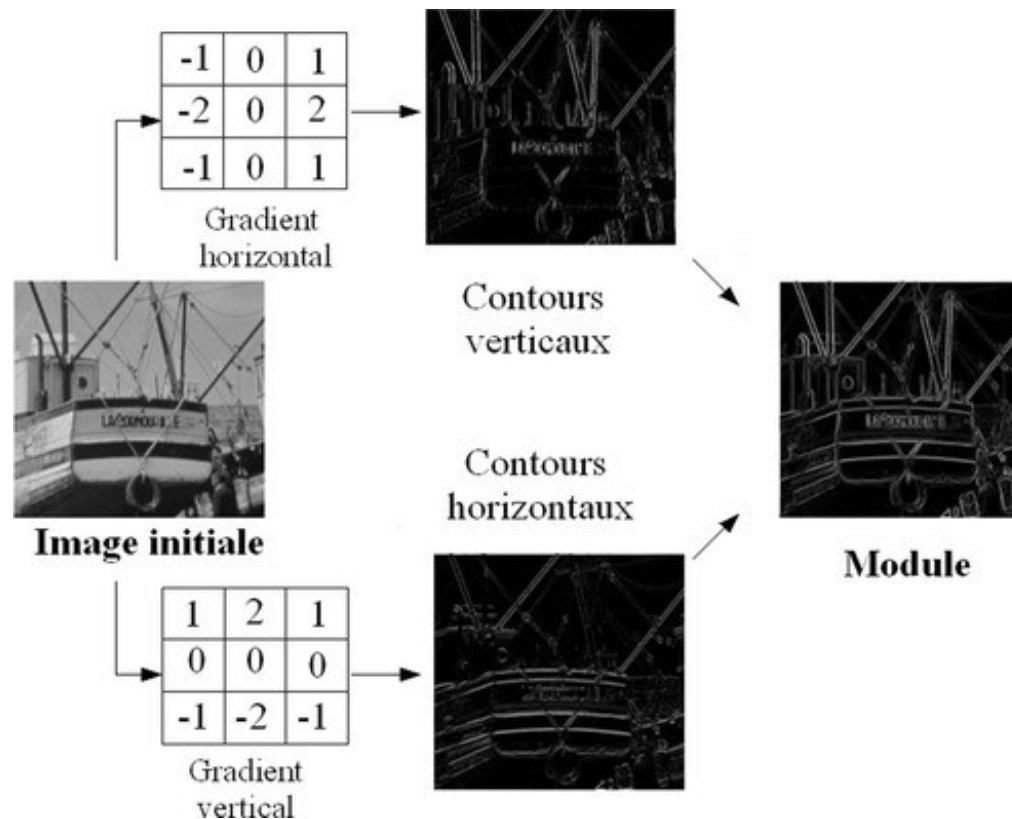
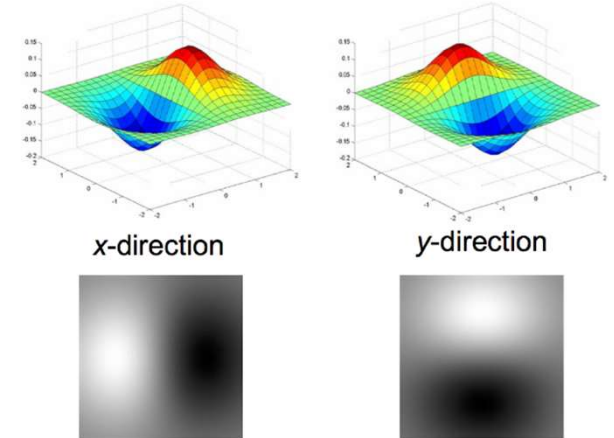


Multiplication and Summation = 0

- On obtient donc une carte d'activation de ce filtre
 - Feature map

Filtre de Sobel

- Dérivée du filtre Gaussien en x et y →
- 2 filtres discrets de directions



Filtre : détecteur d'arêtes

Filter Builder - Alwaysbusy's Corner

Kernel dimensions

Load Filter

Save Filter

Load picture

Apply Filter

X

Y

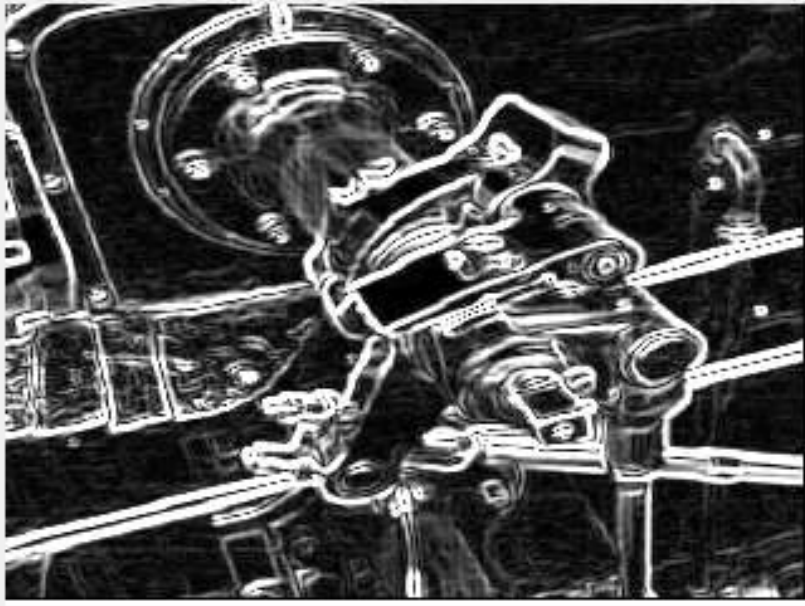

X Kernel

Y Kernel

1	1
2	2
3	3
4	4
5	5

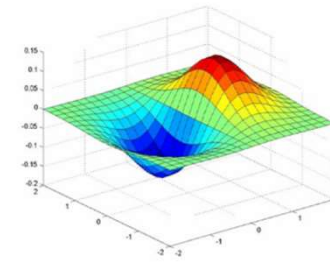
-1	-1	1
-2	0	2
-1	0	1
-2	1	2

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

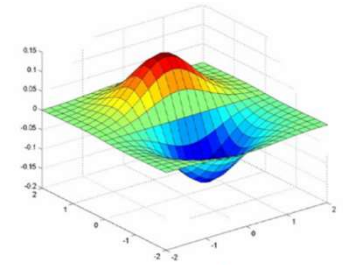
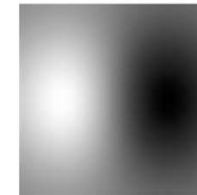


Filtre : Canny edge detector

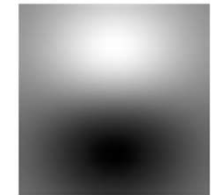
- Dérivée du filtre Gaussien en x et y



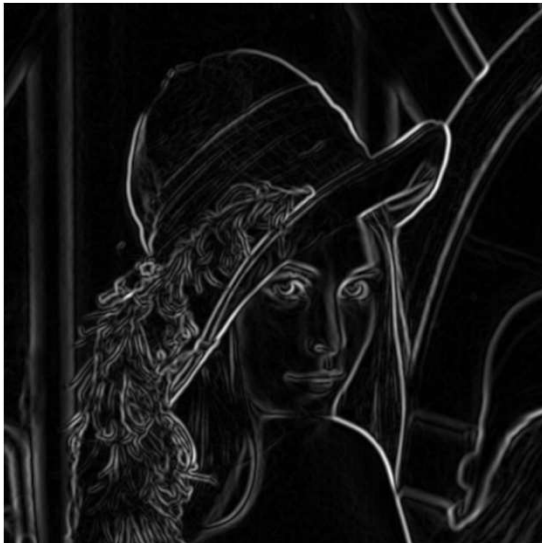
x-direction



y-direction



- Suppression des pixels non-maximaux



Filtre « boussole »

- Détection des 8 directions

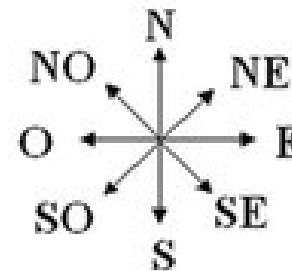
**Exemple :
Robinson (3)**

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

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

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

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



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

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

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

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

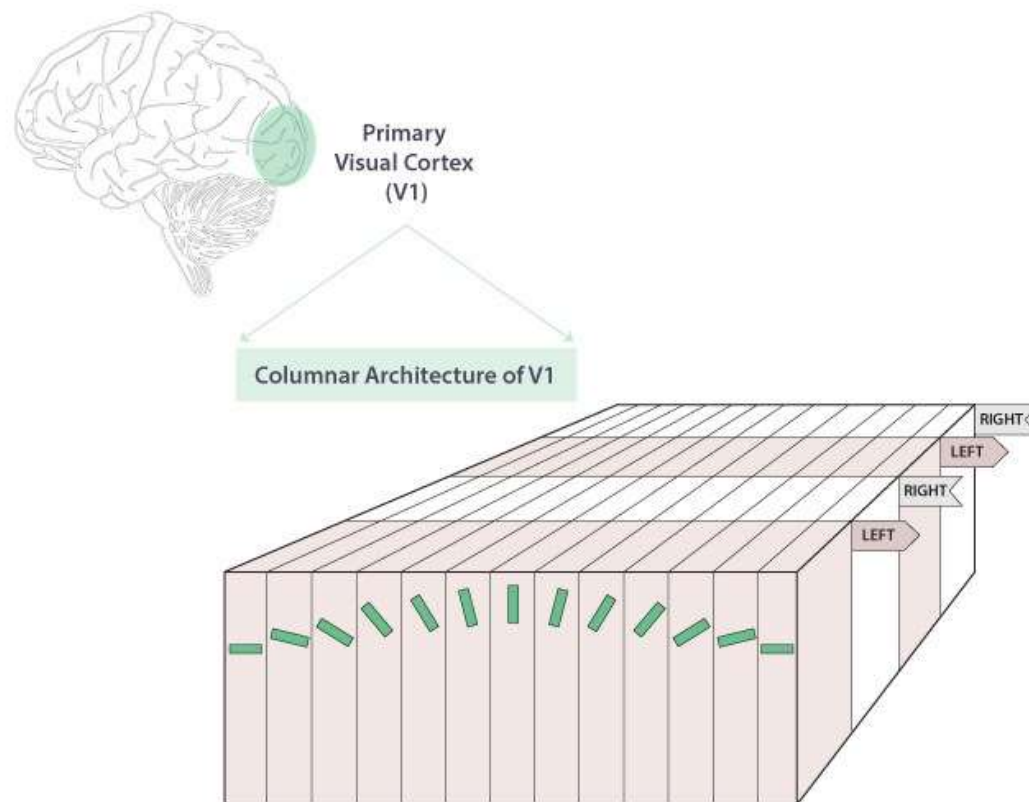
Les Difficultés

Quel enchainement/combinaison de filtres permettront de bien classifier une image ?

- Avant (le deep learning)
 - un humain (ingénieur/chercheur) proposait des filtres (moins de 10) selon son savoir-faire pour produire des descripteurs (quelques centaines de valeurs)
 - Puis classification (SVM, Random Forest, etc.)
- Maintenant, réseau de convolution **ConvNeuralNET (CNN)**
 - Optimise des poids dans plusieurs filtres (plusieurs dizaines/centaines) pour produire les descripteurs (*feature maps*)
 - Puis les dernières couches complètement connectées font la classification

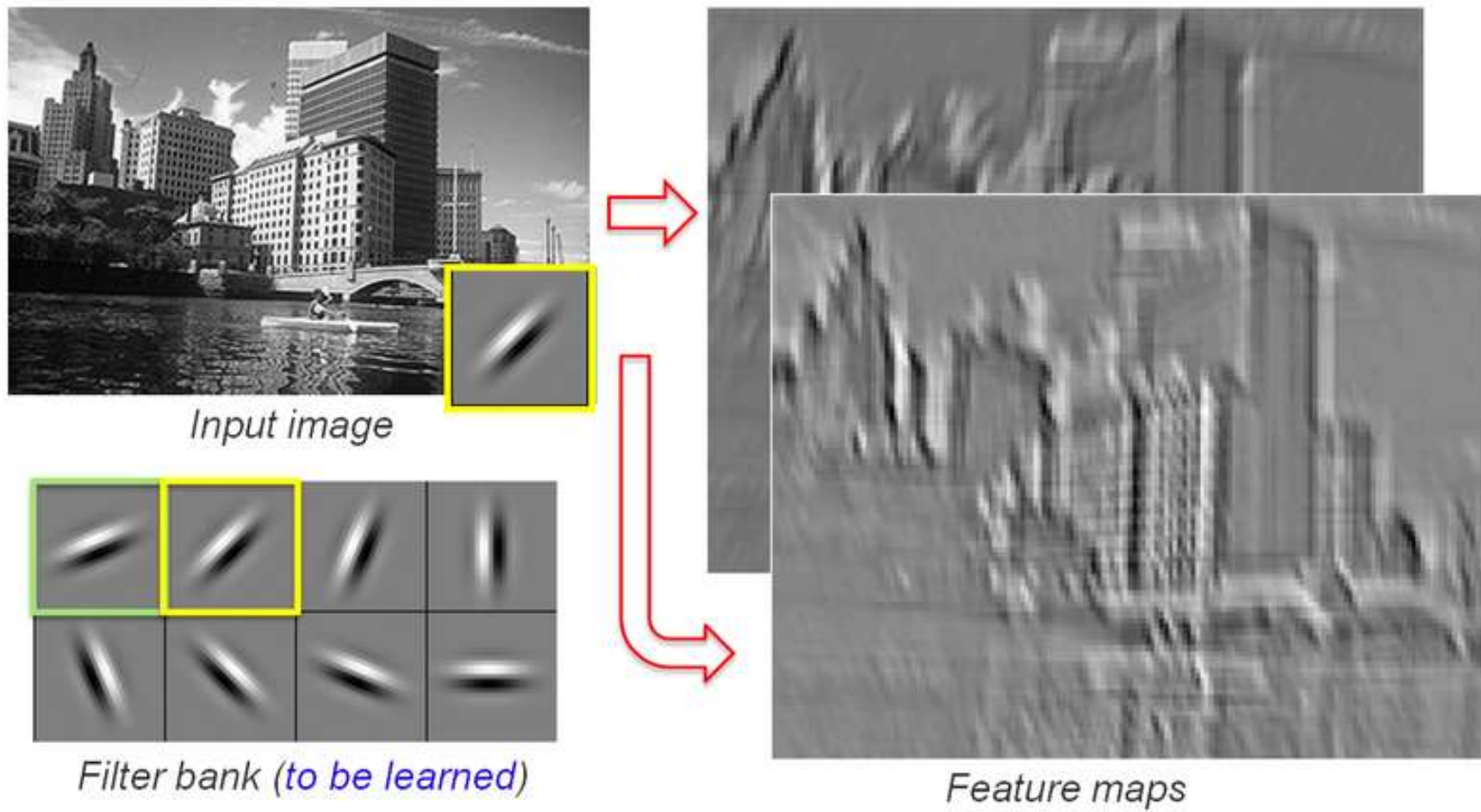
Expérience de Huble

- Huble and Wiesel en 1962 montre que dans le cortex visuel les neurones sont organisés pour répondre à des patrons précis : lignes avec différentes inclinaisons, etc.



Réseau de convolution ConvNET

- Appliquer une banque de N filtres sur une image
 - Donne N images résultats
 - Apprendre ces filtres ?



CNN exemple

- Une convolution (filtre) 3x3 sur une image 6x6x1

1 _{x1}	1 _{x0}	1 _{x1}	0	0
0 _{x0}	1 _{x1}	1 _{x0}	1	0
0 _{x1}	0 _{x0}	1 _{x1}	1	1
0	0	1	1	0
0	1	1	0	0

Image

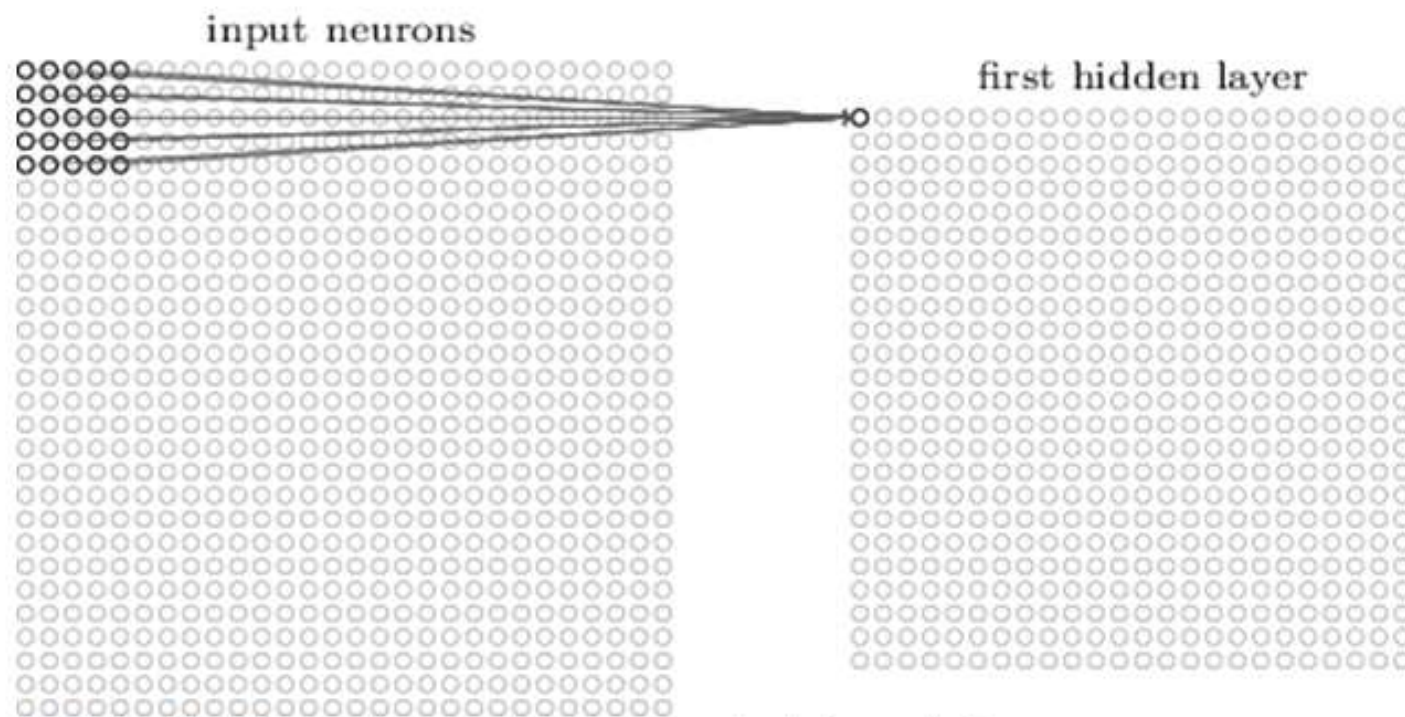
4		

Convolved
Feature

Une convolution

Activation map
Feature map
Image de caractéristiques

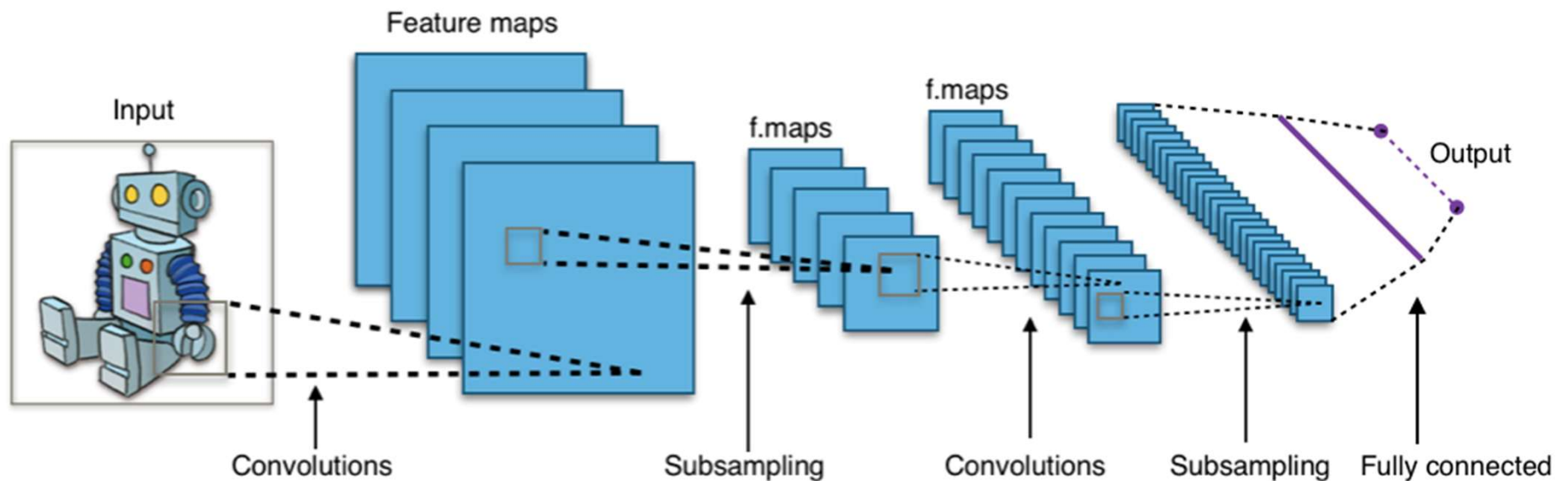
- Input : $32 \times 32 \times 1 \rightarrow \text{Conv}(5,5) \rightarrow 28 \times 28 \times 1$
- Input : $32 \times 32 \times 3 \rightarrow \text{Conv}(5,5,3) \rightarrow 28 \times 28 \times 1$
 - La convolution se fait sur les 3 channels



Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

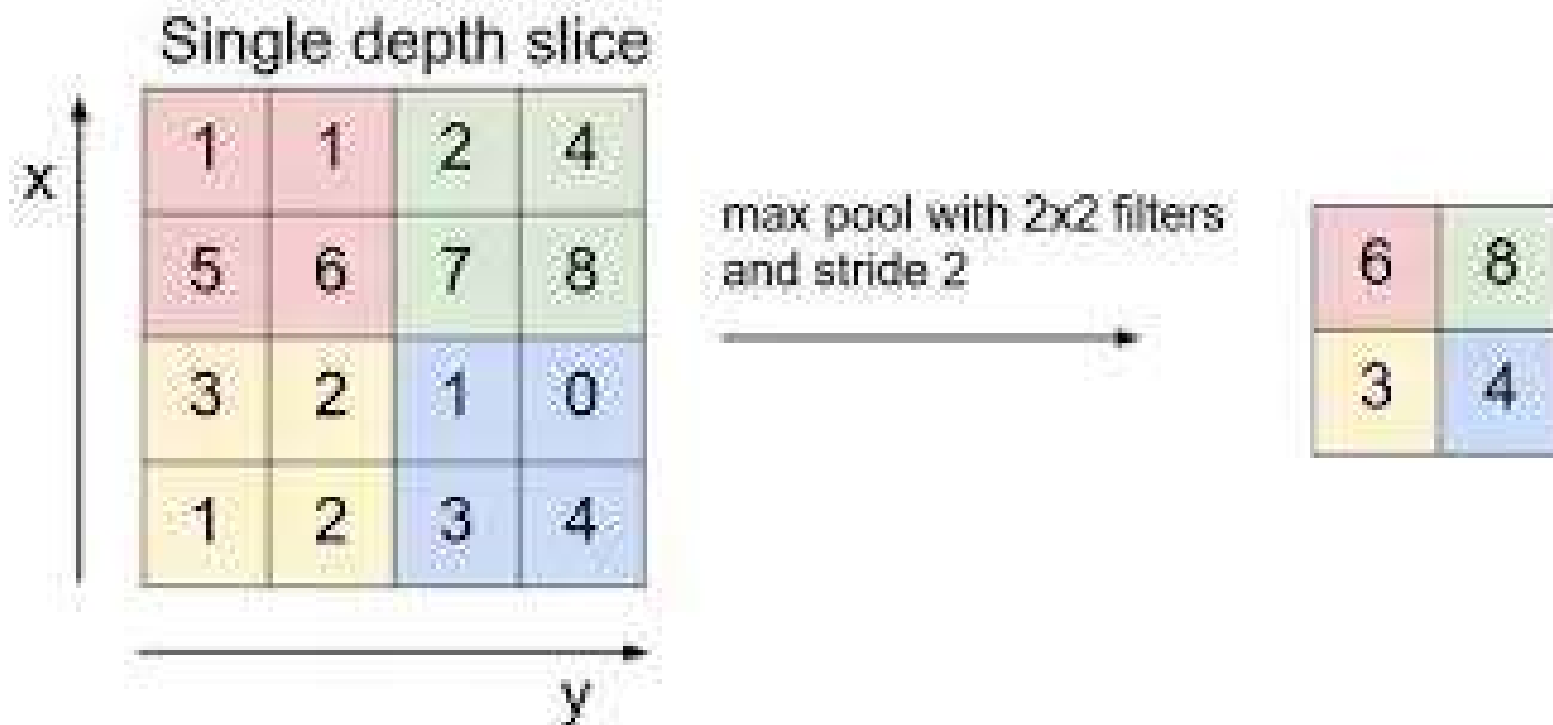
Réseau de convolution ConvNET

- Reconnaitre un objet avec un réseau de convolution



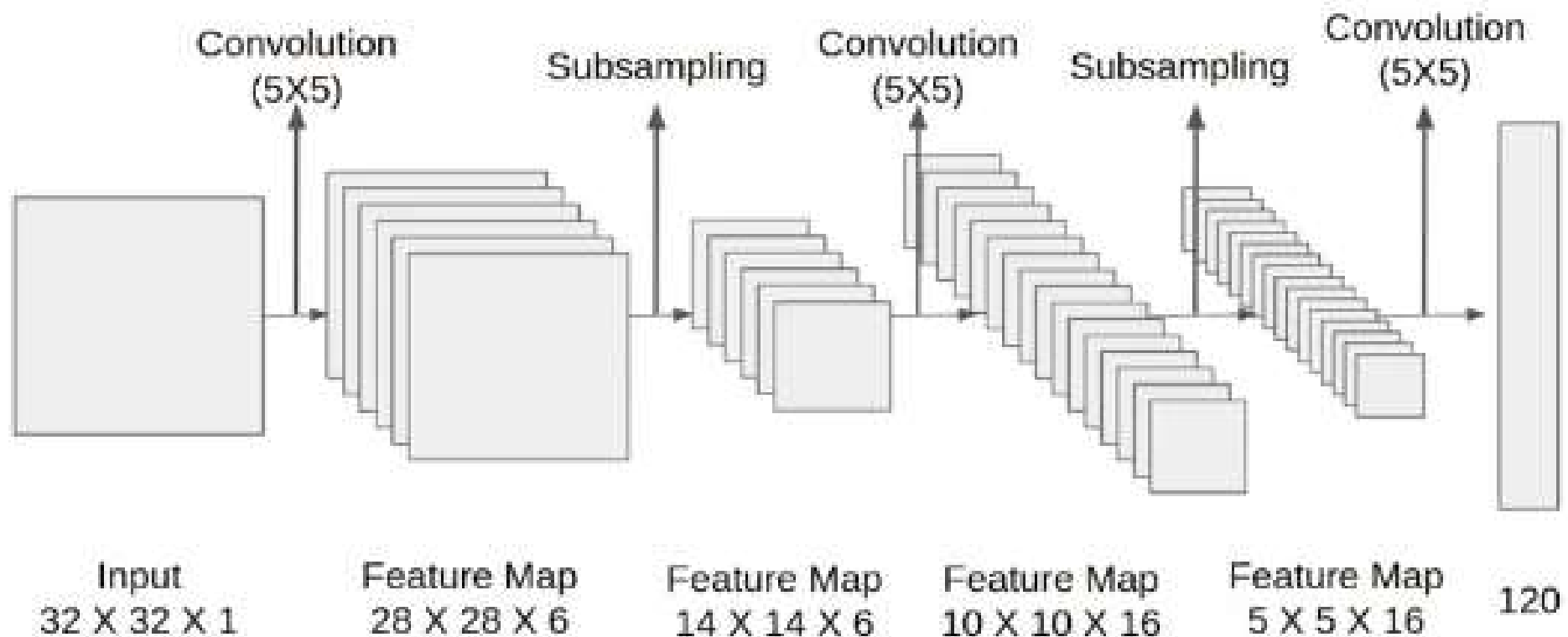
Max Pooling

- Reduction de dimensions



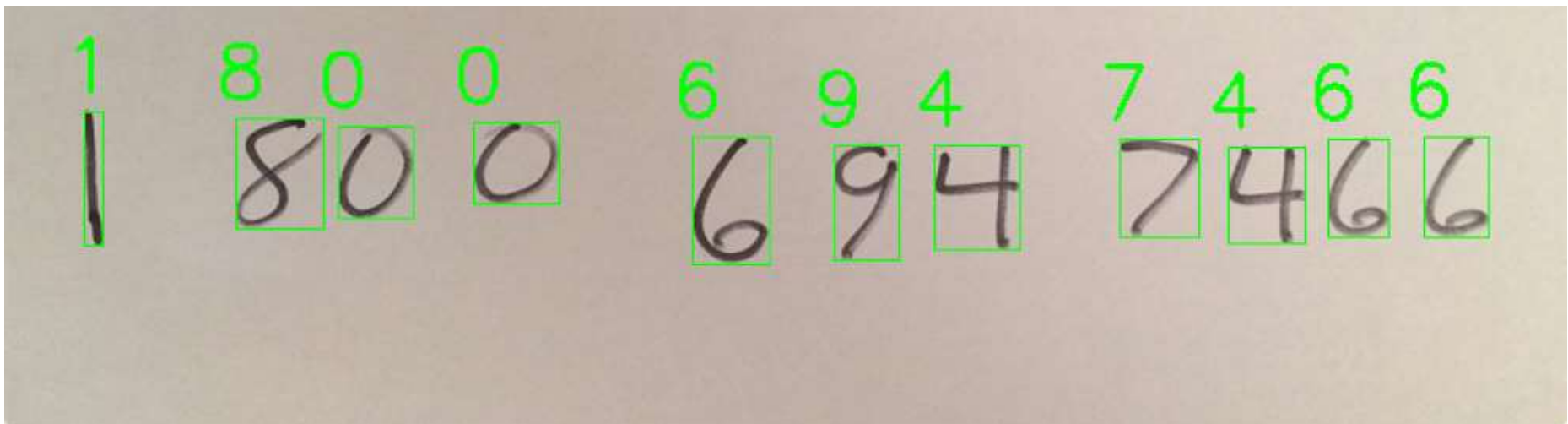
CNN exemple

- Un bon réseau pour commencer à apprendre



Reconnaissance d'écriture

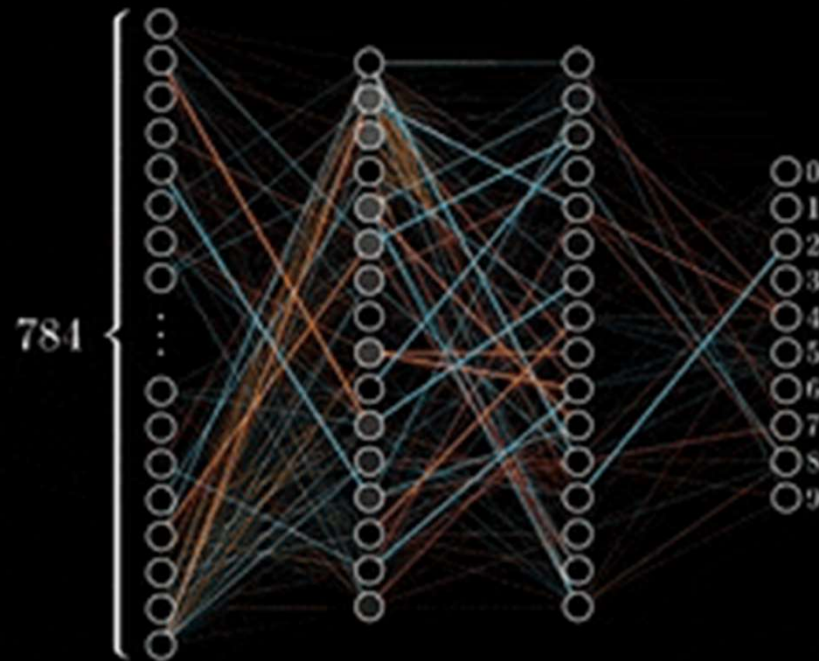
- Reconnaissance d'écriture
 - La Poste : codes postaux sur enveloppe
 - Puis écriture sur tablette
 - Le problème classique pour apprendre les CNN
 - Base de données de chiffres manuscrits : MNIST
- Avec un ConvNET, taux de bonne reco > 99%



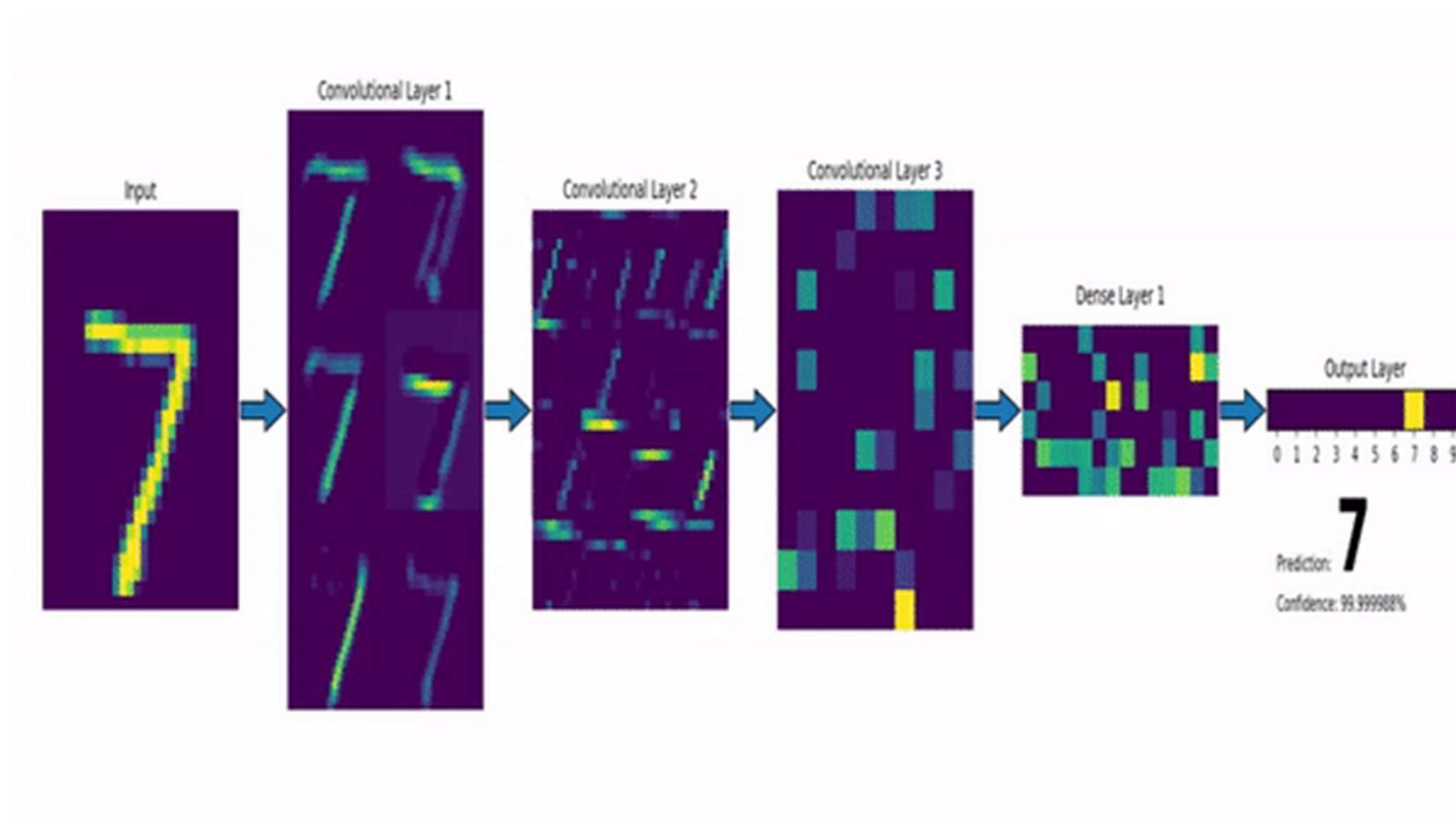
CNN : entrainement

Training in
progress...

 $\rightarrow 9$



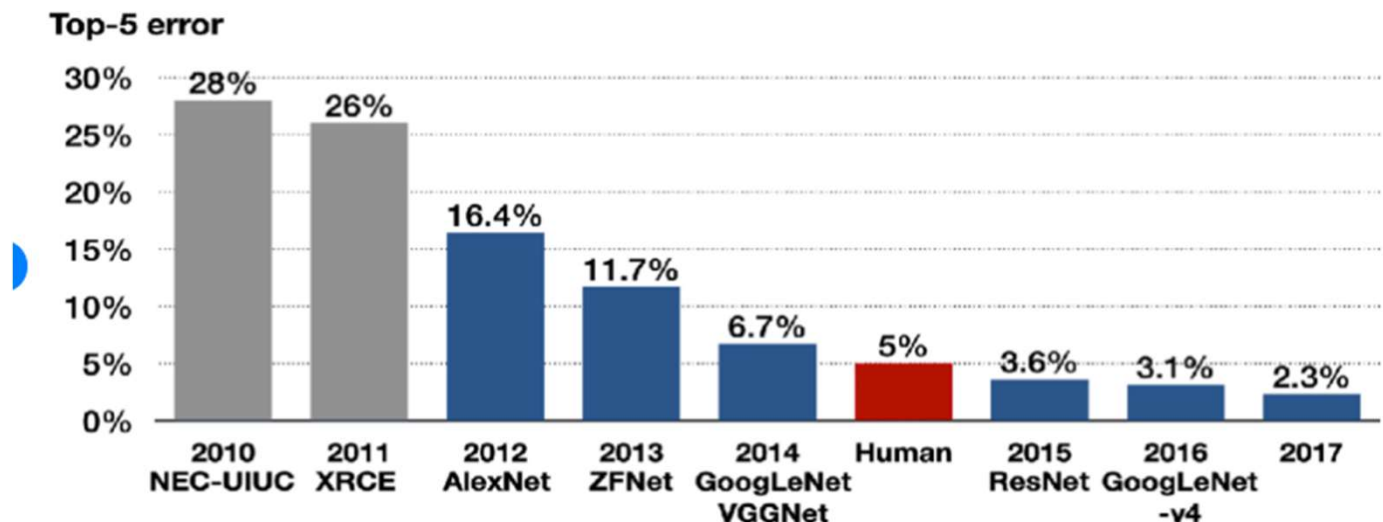
CNN example



IMAGENET=corpus d'images

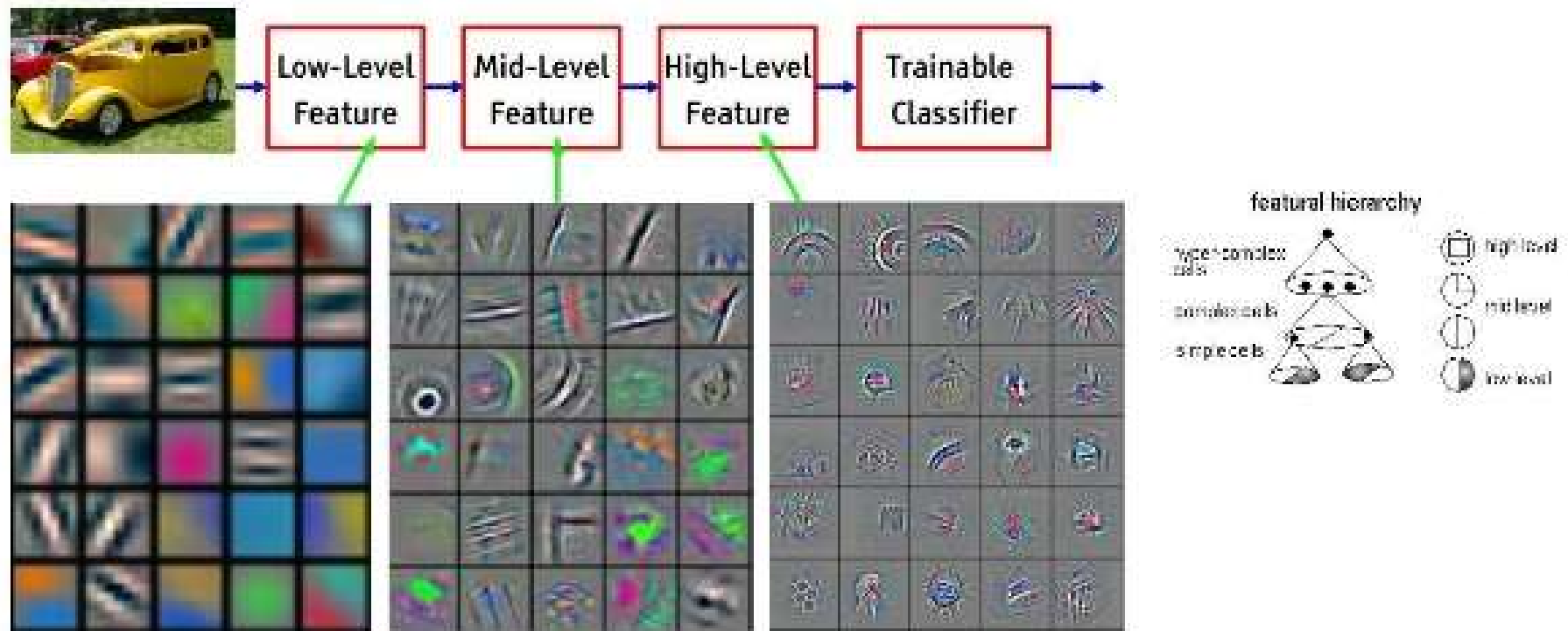


- Classification
 - Historique vision humaine : repérer un prédateur ou un membre de sa famille rapidement
 - Concours IMAGENET → mettre un label sur une image
 - 14 millions d'images avec 20000 labels
 - En bleu, méthodes à base de réseaux



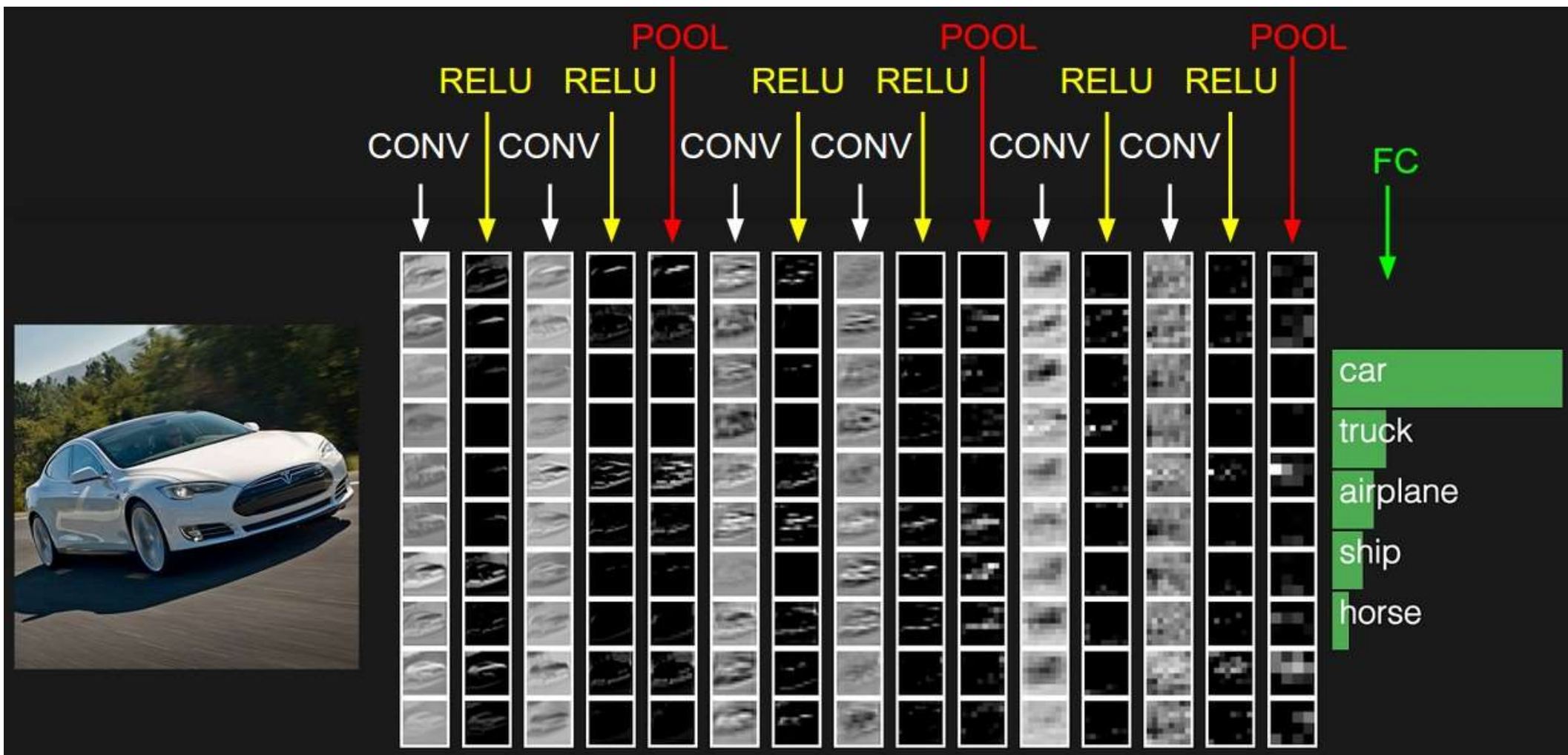
Réseau de convolution ConvNET

ConvNet : Interpretation



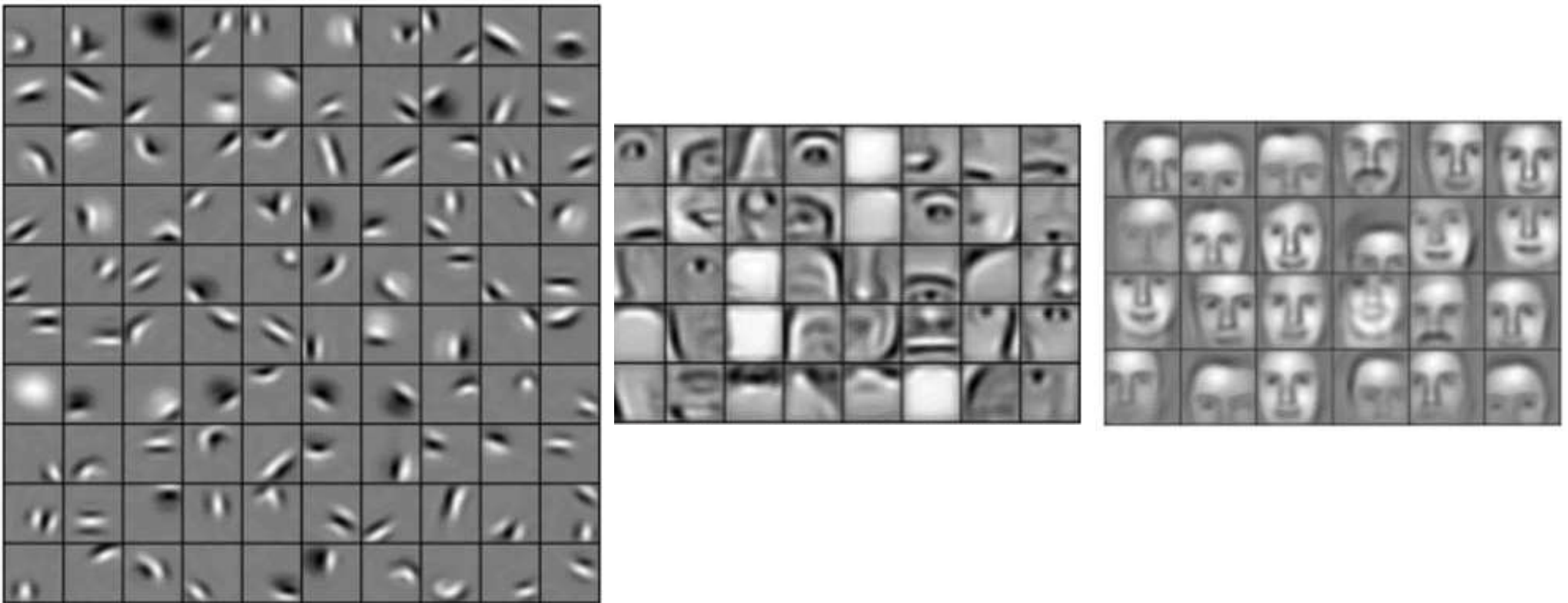
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Réseau de convolution ConvNET



ConvNET : visage

- Exemple de features (filtres) produit par le réseau dans les couches cachées

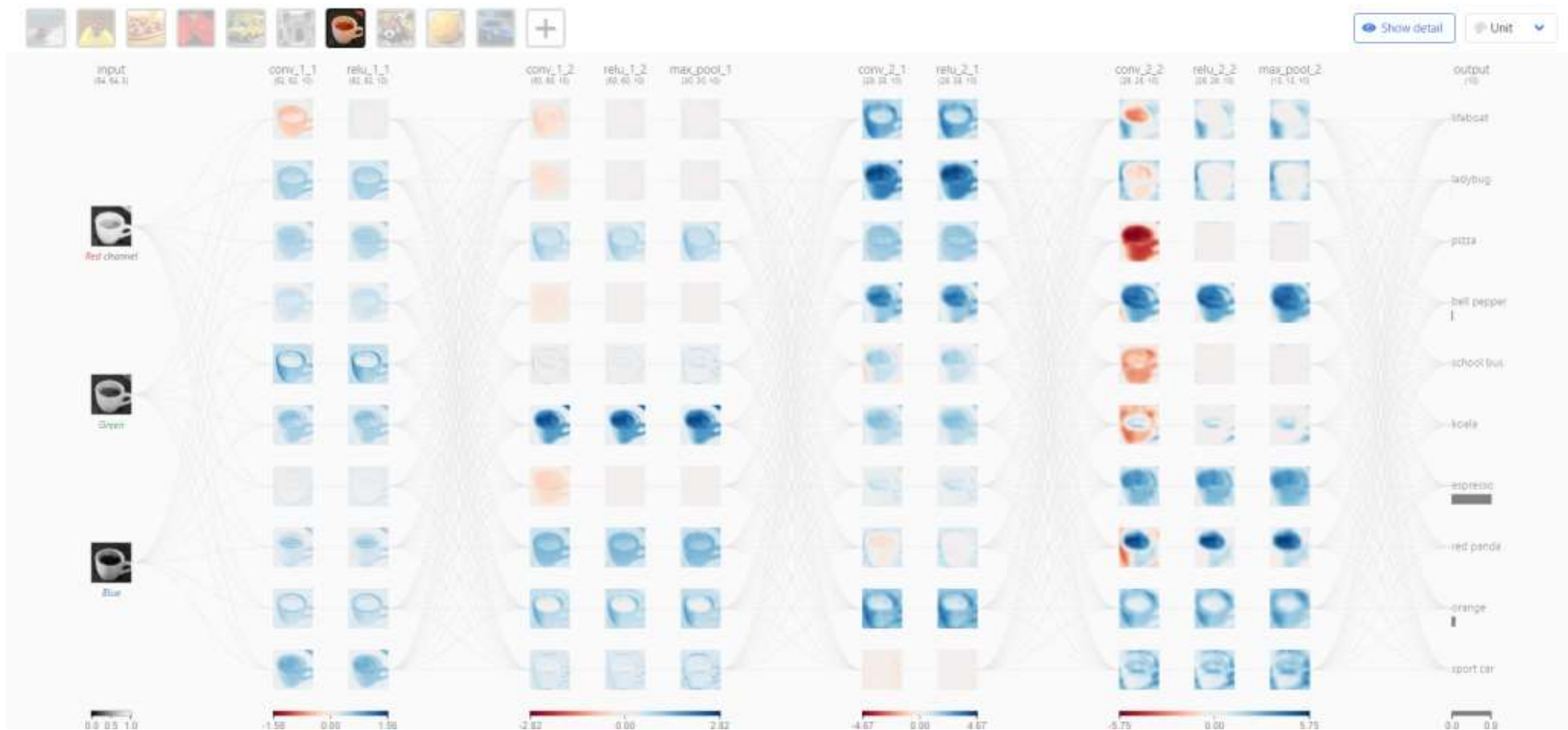


Couches du début

Couches profondes

CNN en démo

- <https://poloclub.github.io/cnn-explainer/>



VGG

K. Simonyan, A. Zisserman

[Very Deep Convolutional Networks for Large-Scale Image Recognition](#)

arXiv technical report, 2014

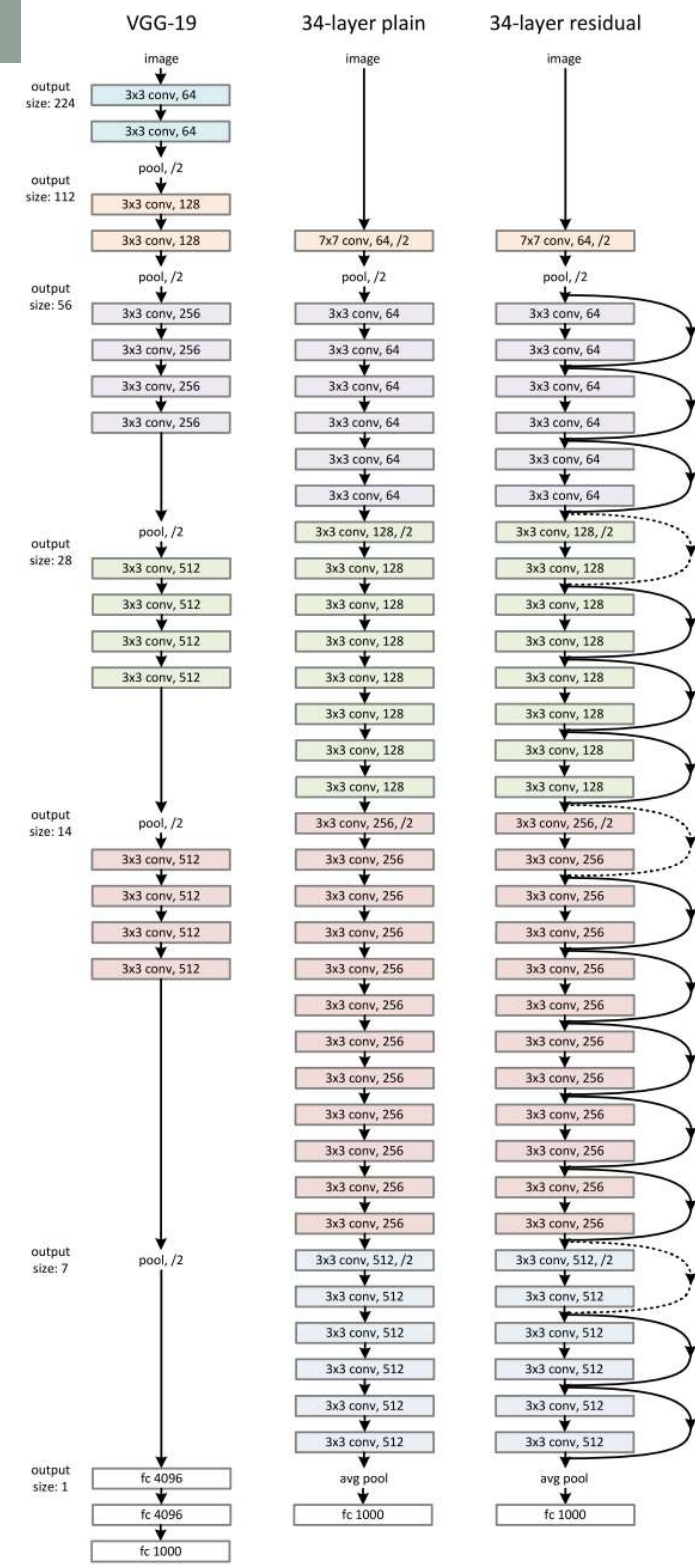
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	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					

VGG19 : voir TP pour afficher les couches

```
0  Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
1  ReLU(inplace)
2  Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
3  ReLU(inplace)
4  MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
5  Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
6  ReLU(inplace)
7  Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
8  ReLU(inplace)
9  MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
10 Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
11 ReLU(inplace)
12 Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
13 ReLU(inplace)
14 Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
15 ReLU(inplace)
16 Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
17 ReLU(inplace)
18 MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
19 Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
20 ReLU(inplace)
21 Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
22 ReLU(inplace)
```

ResNET-50

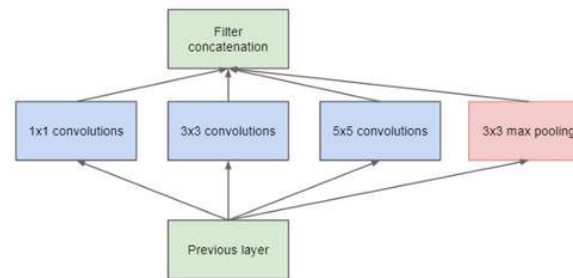
Deep Residual Learning for Image Recognition
Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun
<https://arxiv.org/abs/1512.03385>



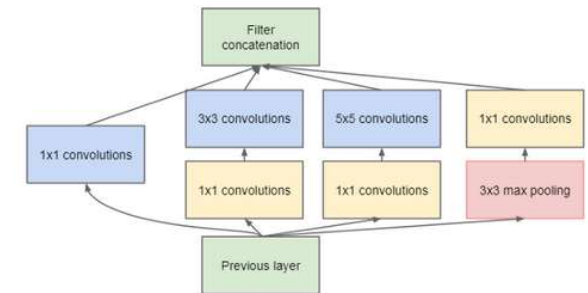
Réseaux de reconnaissance

Souvent utilisés en pré-traitement comme « traitement d'images »

- VGG
 - VGG16, VGG19
 - Série de Conv, Max-pooling, and activation, puis fully-connected (FC)
- ResNet50
 - Grande profondeur
- Inception v3
 - Large
- Xception
 - extreme inception
- MobileNet
 - Optimisé pour mobile





(a) Inception module, naïve version



(b) Inception module with dimension reductions

- Dans KERAS / PyTorch ces modèles sont pré-entraînés (ouf)

Les données

	VGGNet	DeepVideo	GNMT
Used For	Identifying Image Category	Identifying Video Category	Translation
Input	Image 	Video 	English Text 
Output	1000 Categories	47 Categories	French Text
Parameters	140M	~100M	380M
Data Size	1.2M Images with assigned Category	1.1M Videos with assigned Category	6M Sentence Pairs, 340M Words
Dataset	ILSVRC-2012	Sports-1M	WMT'14

Number of parameters (in millions), for popular neural networks.

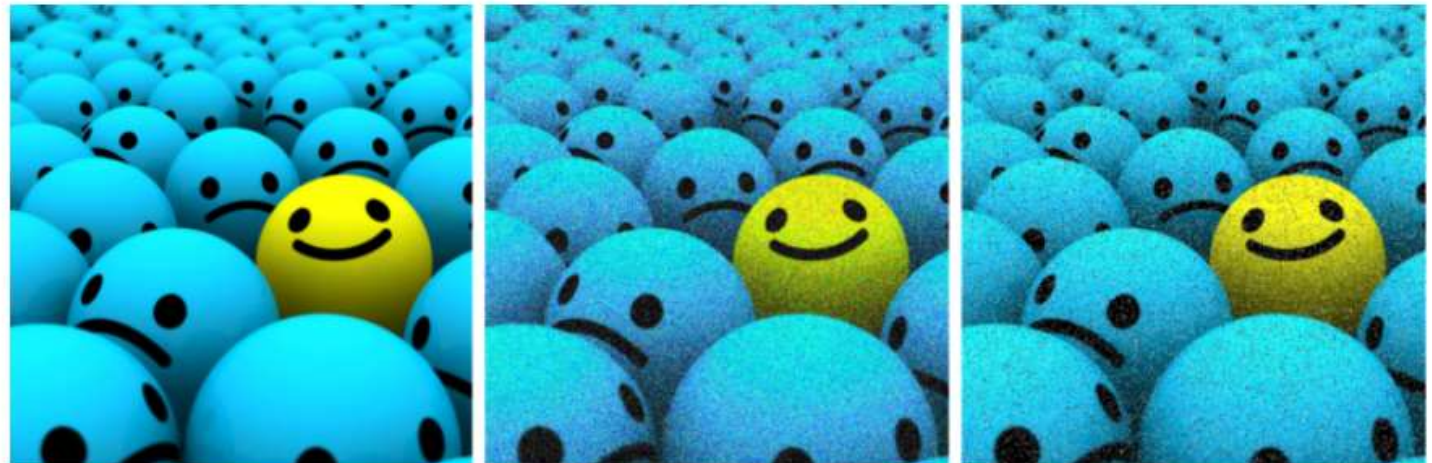
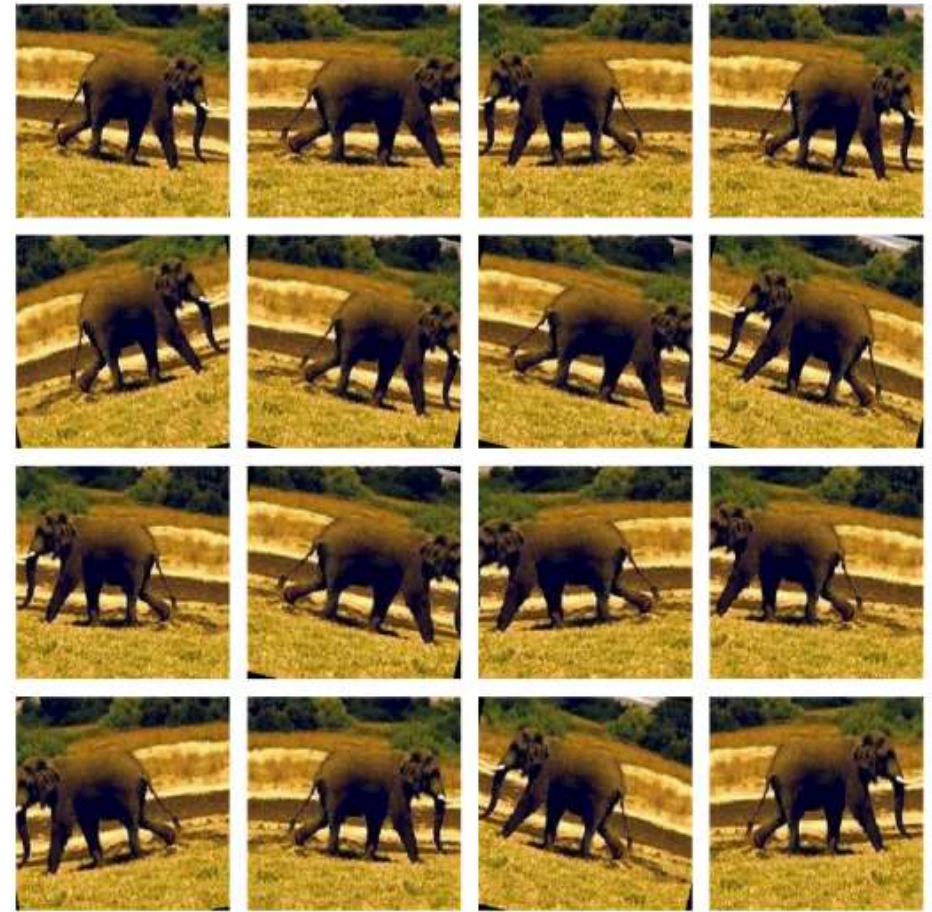
Problème : manque de données pour DL

- Taille de certaines base de données
 - MNIST : nombres manuscrits, 70000 images
 - EMNIST : lettres manuscrites
 - Équilibré 134000 images
 - Sinon 814000 images
 - ImageNET : 14 millions d'images avec 20000 labels
 - CelebFaces (CelebA) : 202,599 images de visages de 10,177 célébrités
- Pour de nombreux autres problèmes
 - Cohn Kanade : 486 vidéo de 97 personnes exprimant une expression
 - ...
- Branches du machine learning qui cherchent à fonctionner avec moins de données
 - DL plus efficaces : GAN, etc.
 - One-shot learning / Few-shot learning

Augmentation de données

La base

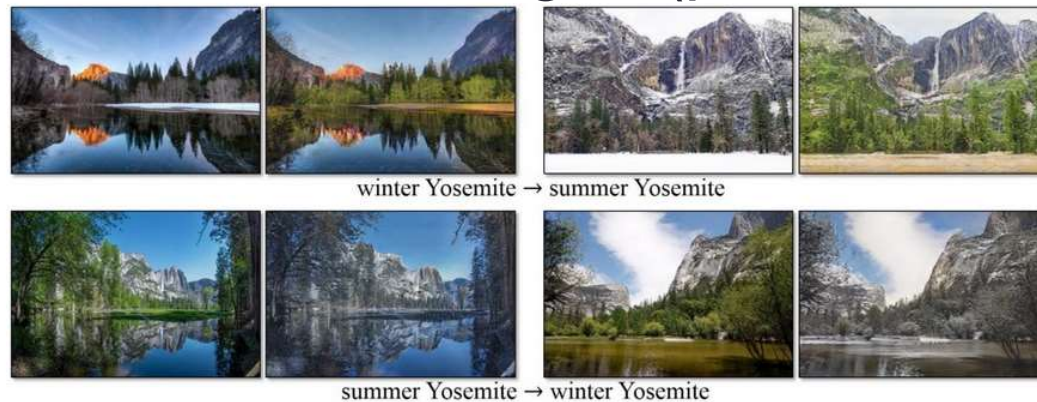
- Flip
- Rotation
- Scale
- Crop
- Translation
- Gaussien Noise



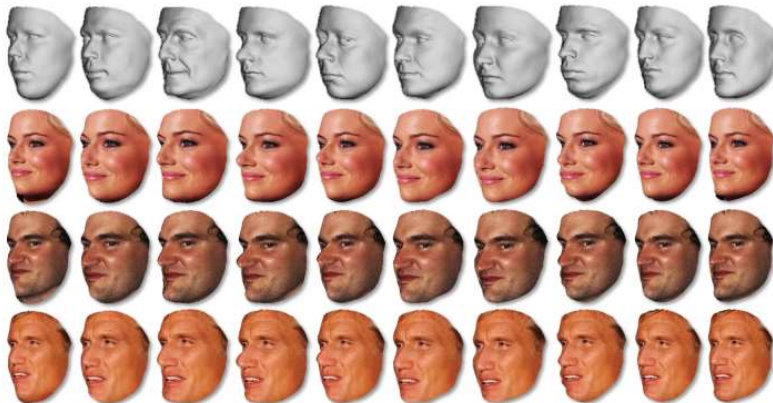
Augmentation de données

Un peu plus loin

- GAN pour transformer des images (palette de couleur, style, etc.)



- Données issus d'images de synthèses



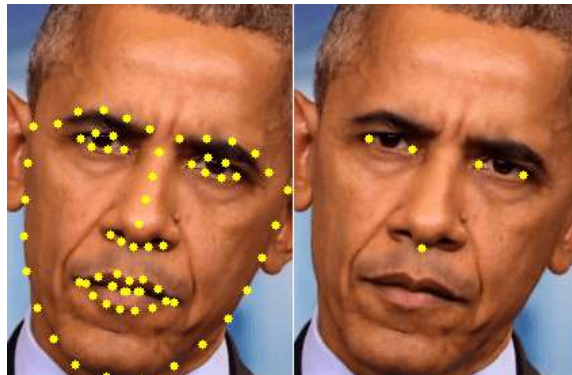
Do We Really Need to Collect Millions of Faces for Effective Face Recognition?

<https://arxiv.org/pdf/1603.07057.pdf>

Augmentation de données : aider le réseau avec des données annexes

Par exemple : corpus d'images/vidéo de visages

- DL sur uniquement les images, possible mais ...
 - Possibilité d'en extraire des points du visage par détection : coins de la bouche, nez, yeux
 - DL sur les points caractéristiques + images
- Bien plus efficaces !!!



reconnaissance



Expressions
Nom de la personne
Etc.

Avantages de cette évolution « deep »

On profite

- Framework
 - GPU, Optimiseur, etc.
 - Learning en python mais après utilisation en C++, C#, Java, etc.
 - Format standard de fichiers : Open Neural Network Exchange
- Communauté grande
 - Nombreux tutoriaux et explications
 - medium.com
 - letslearnai.com
 - Etc.
- Recherche reproductible (Github)

TP AVEC CNN

- Classifier → voir la page web de l'UE
- Appliquer les CNN à transférer le style d'une image sur une autre

TRANSFERT DE STYLE ENTRE IMAGES

1 Upload photo

The first picture defines the scene you would like to have painted.



2 Choose style

Choose among predefined styles or upload your own style image.



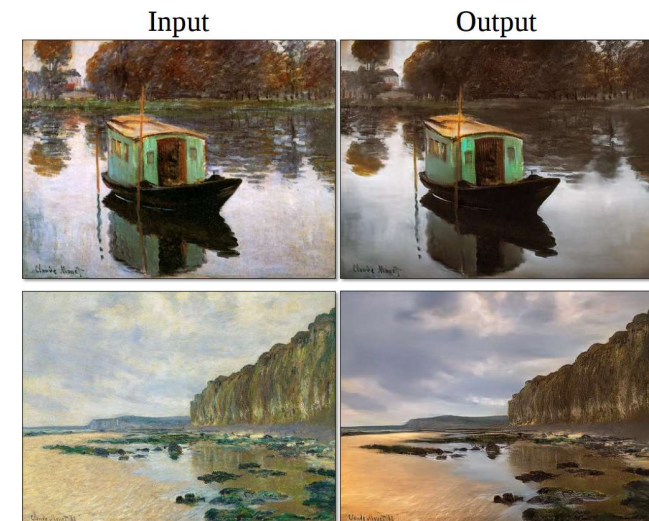
3 Submit

Our servers paint the image for you. You get an email when it's done.



Transfert de style entre images

- Toutes les qualités d'un bon TP
 - Utilisation des frameworks de DL
 - Optimisation
 - Utilisation des réseaux, mais sans avoir besoin de les entrainer (temps de calcul raisonnable pour un TP)



Transfert de style entre images

- Différentier
 - Contenu de l'image : objets et leurs places/positions/orientations
 - Style : couleur et textures
- VGG19 pour extraire les caractéristiques
 - Chaque couche de convolutions va produire une carte de caractéristiques (features)
 - Optimisation avec deux termes : $\text{Coût_Contenu} + \text{Coût_Style}$



Caractéristiques à différentes échelles

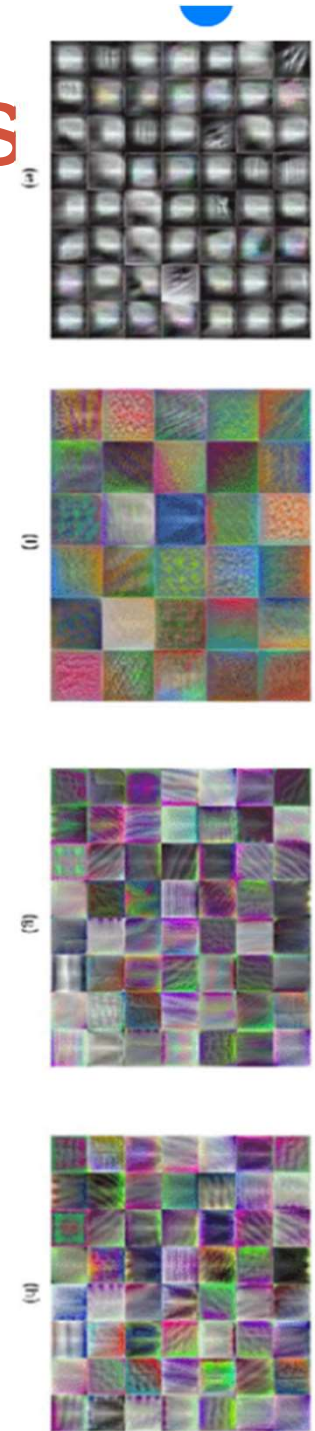
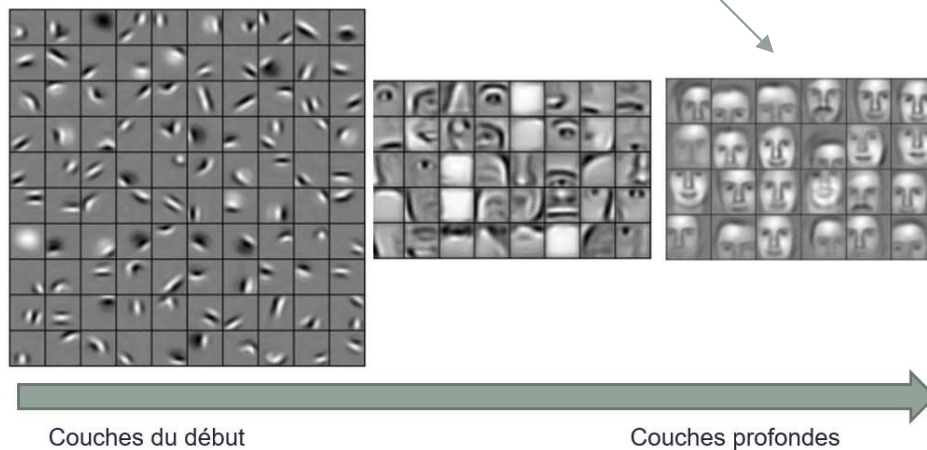
Transfert de style entre images

VGG19 pour extraire les caractéristiques

- Chaque couche de convolutions va produire un vecteur de caractéristiques de dimension

Batch_size x Nfeatures x Height x Weight

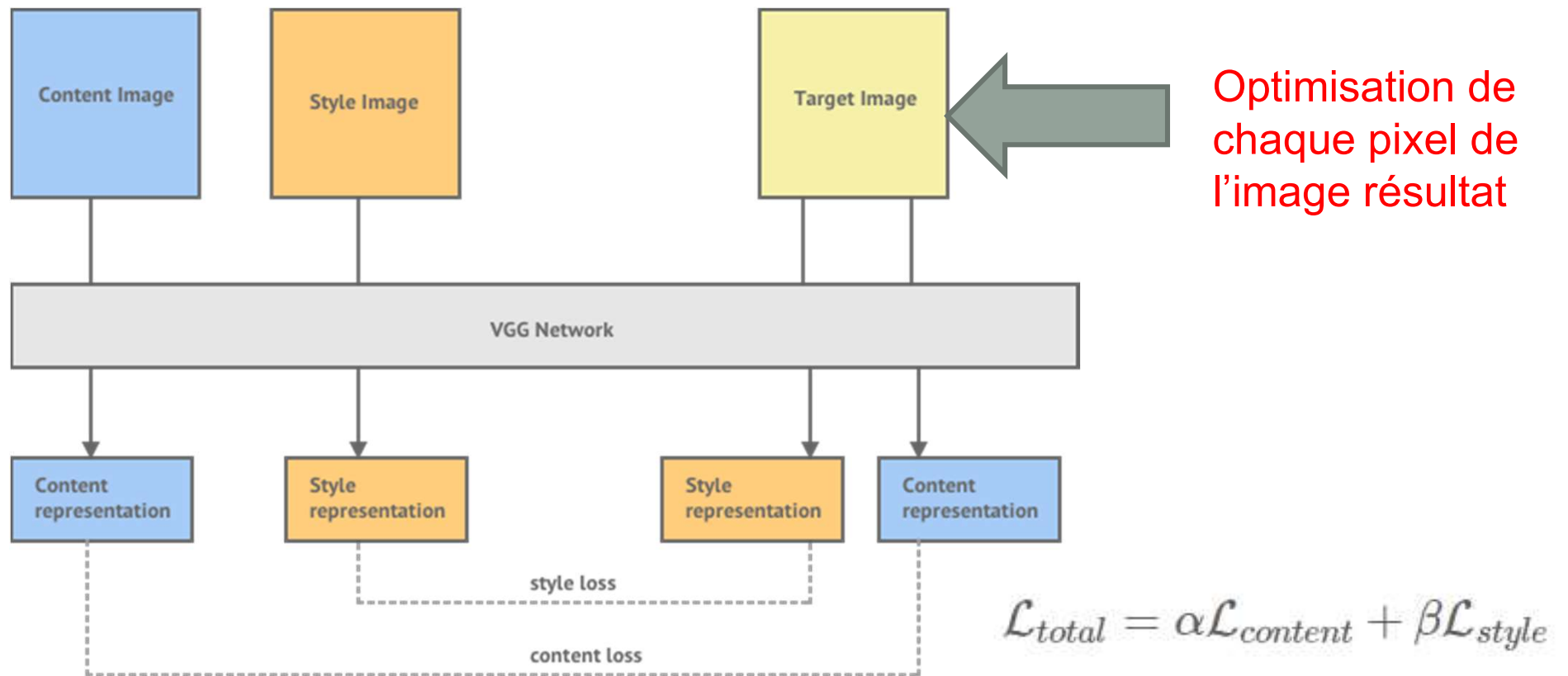
- Certaines couches codent plutôt le contenu (vers le fond du réseau), d'autres plutôt le style (vers le début)



Features visualization of VGG network

Transfert de style : optimisation

- Il n'y a pas d'optimisation des neurones d'un réseau.
- Le réseau (VGG) est utilisé pour produire les descripteurs (features)
- Le framework de Deeplearning (pytorch) est utilisé pour optimiser les pixels de l'image



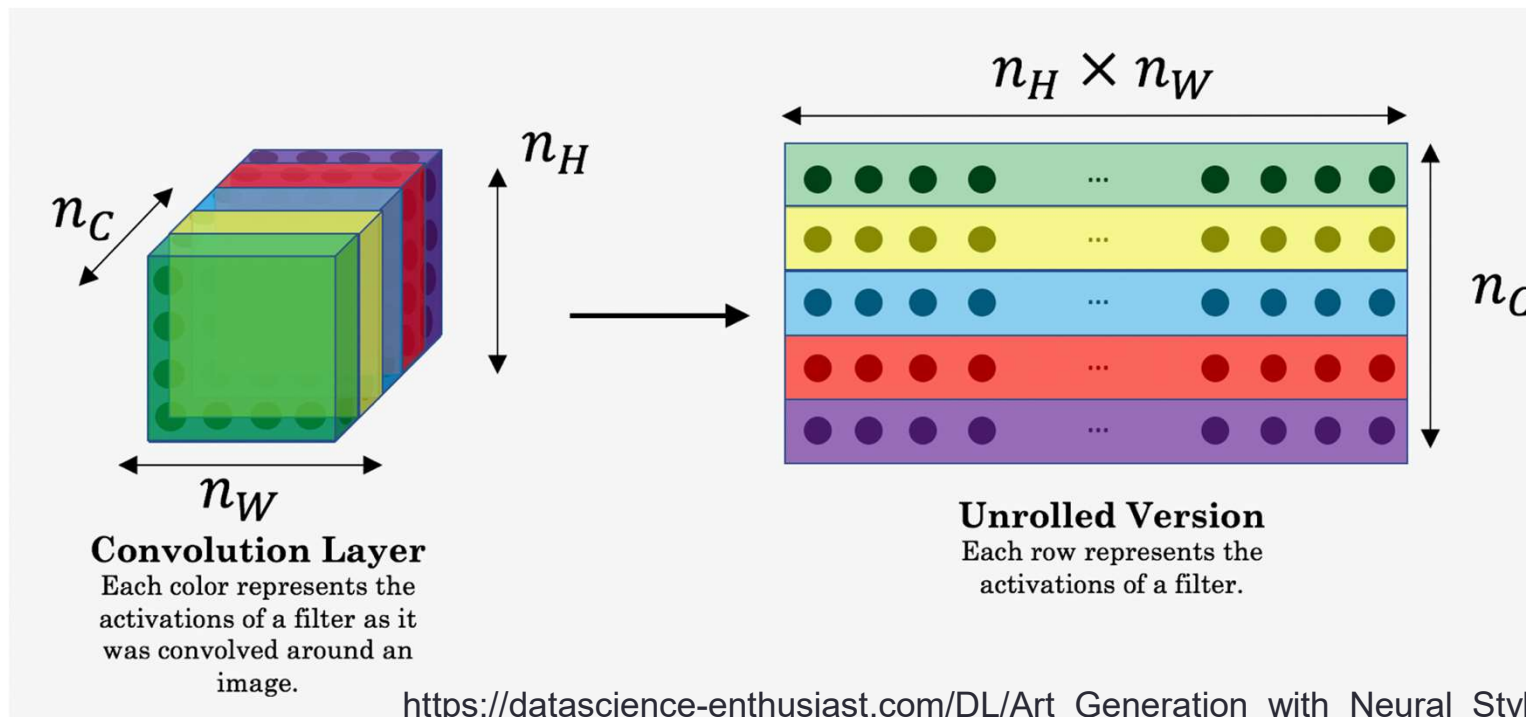
Transfert de style entre images

VGG19 pour extraire les caractéristiques

- Chaque couche de convolutions va produire un vecteur de caractéristiques de dimension

$N_features(N_c \text{ sur la figure}) \times \text{Height} \times \text{Weight}$

→ à aplatir en $N_features \times N_pixels$ avec $N_pixels = n_height \times n_weight$



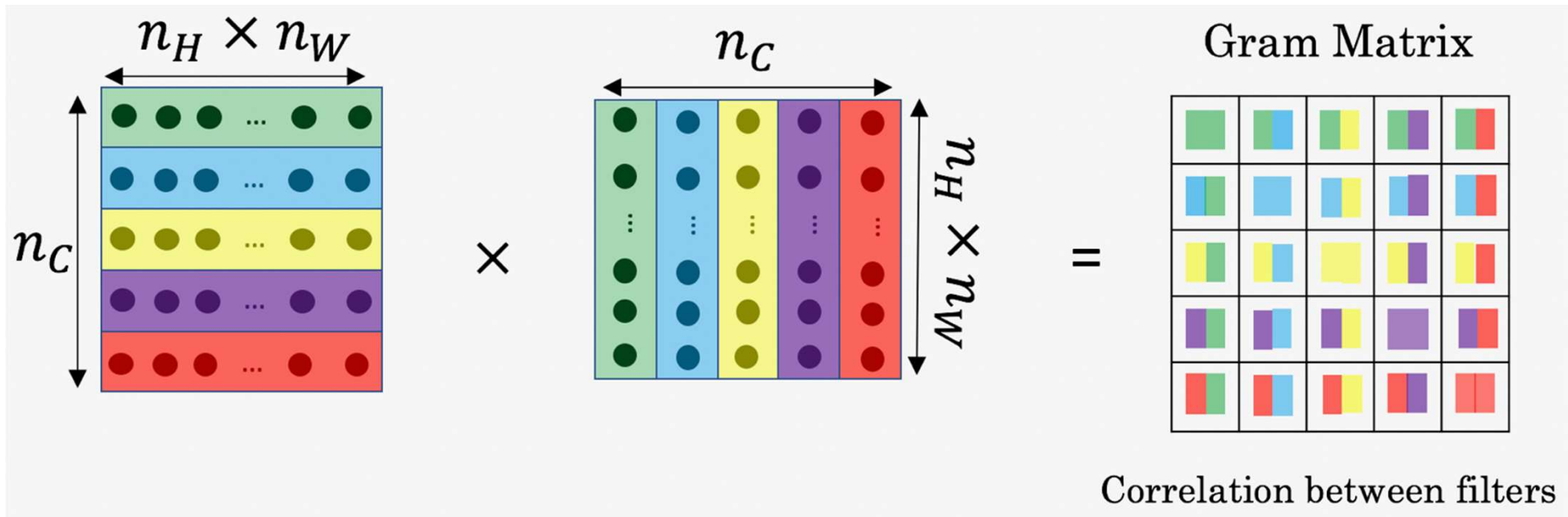
Transfert de style : matrice de Gram

- **Matrice de Gram : $M \times M^t$**

- Produit scalaire entre toutes les caractéristiques
- Corrélation entre les features
- Avec une matrice F de caractéristique, une entrée de la matrice de Gram G est le produit scalaire entre 2 caractéristiques

$$G_{ij} = \sum_k F_{ik} F_{jk}$$

Transfert de style : matrice de Gram



Si une entrée dans la matrice de Gram a une valeur proche de 0, cela signifie que les 2 *features* ne s'activent pas simultanément (non corrélation). Et vice versa, si une entrée a une grande valeur, cela signifie que les 2 *features* s'activent simultanément (corrélation).

Nous allons chercher à créer une image qui réplique un même schémas d'activations des *features* de style.

Transfert de style : coût de contenu

- Si on peut construire une image qui a une carte de caractéristiques équivalentes pour un niveau de convolution donné à une autre image. Ces deux images auront le même contenu (surtout pour les couches profondes) — mais pas nécessairement la même texture ou style.
- Soit une couche de convolution **l** dans VGG, la fonction de coût de contenu est définie comme la moyenne au carré de l'erreur entre la carte de *features* **F** de l'image de contenu **C** et la carte de *features* de l'image générée **Y**.

$$\mathcal{L}_{content} = \frac{1}{2} \sum_{i,j} (F_{ij}^l - P_{ij}^l)^2$$

Transfert de style : coût de style

- Le calcul de coût du style est similaire au calcul de coût de contenu, mais on le calcule à partir de la matrice de Gram à la place de directement utiliser les features

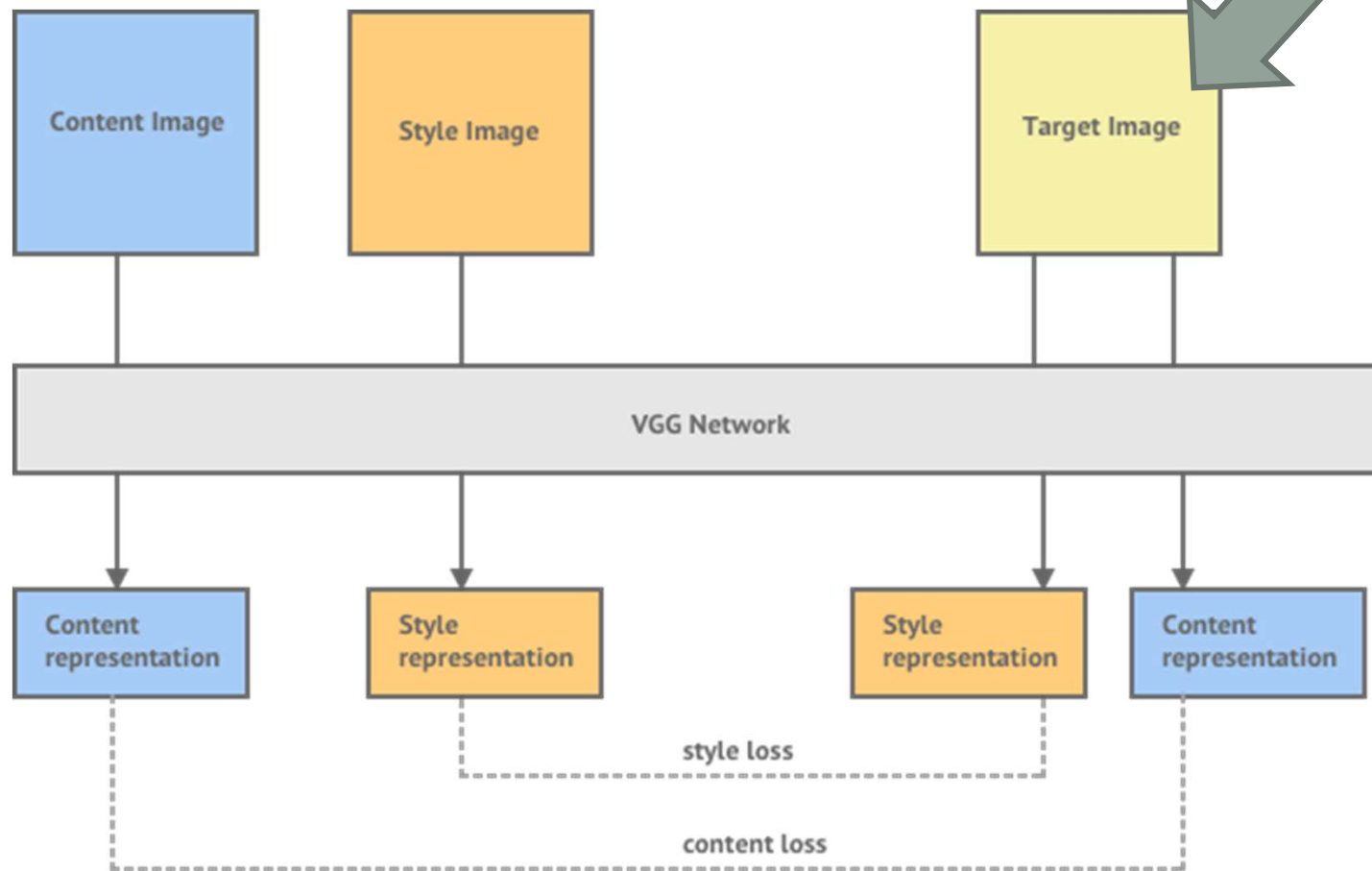
$$\mathcal{L}_{style} = \frac{1}{2} \sum_{l=0}^L (G_{ij}^l - A_{ij}^l)^2$$

Transfert de style : optimisation

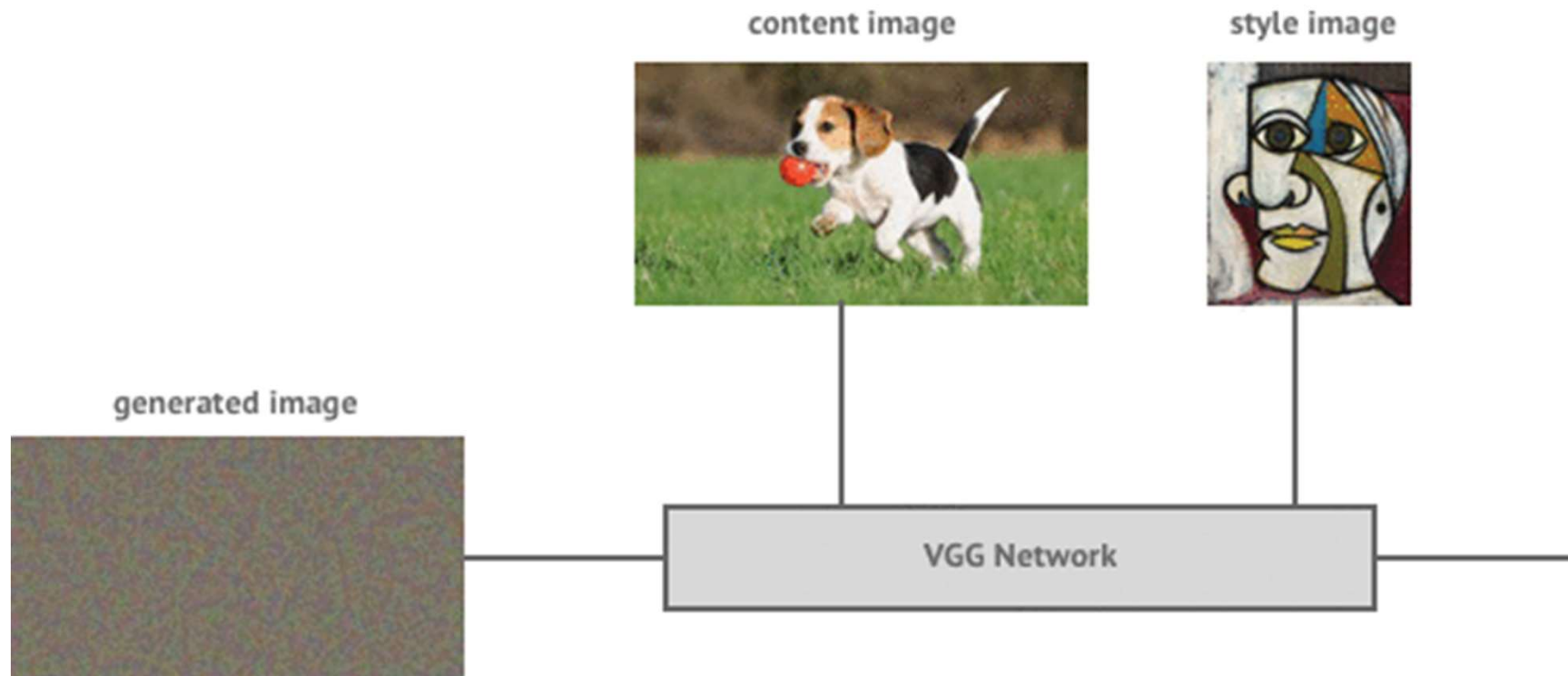
- La fonction de coût à optimiser

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$$

Optimisation de chaque
Pixel de l'image résultat



Transfert de style



Conclusion

- TP
- ...
- De nombreuses approches encore à voir ...

