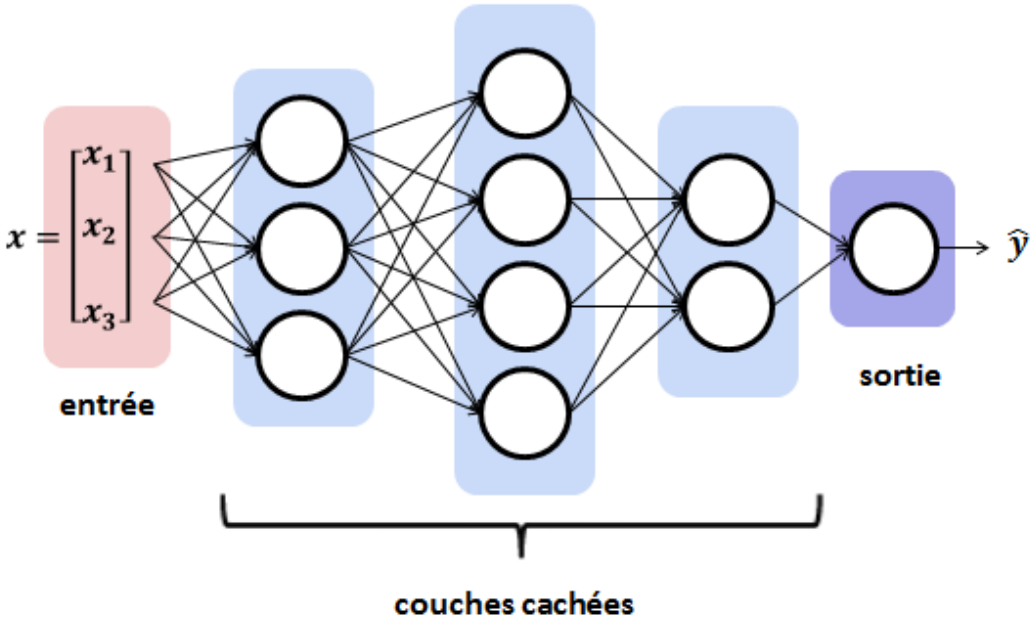
Traitement et Analyse d'Images

Apprentissage profond
Réseaux de neurones convolutionnels

olivier.bernard@insa-lyon.fr

Analyse d'images par réseaux de neurones

- Perceptron multicouches (MLP)



- Vectorisation d'une image en entrée
- Image 256 x 256

$$x^{(i)} \in \mathbb{R}^{[65536\times 1]}$$

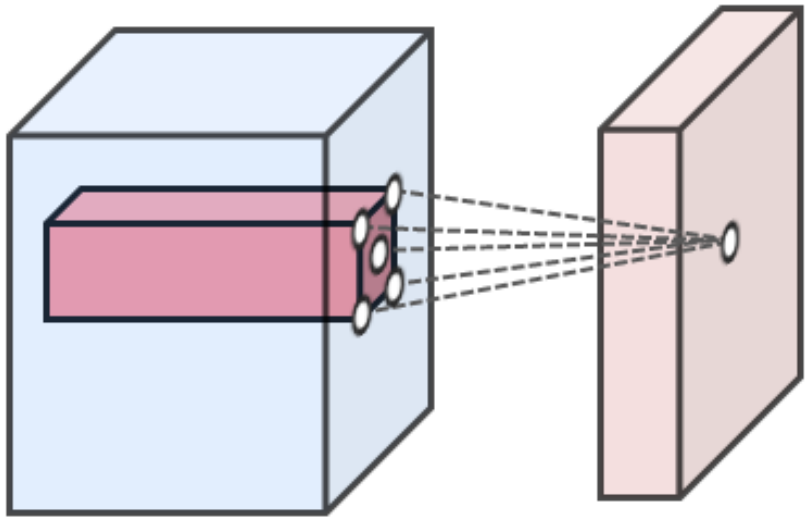
- Réseau à 1 couches cachée avec $n^{[1]} = n_X$

$$\#\, param > 4\, M$$

Beaucoup de paramètres pour des réseaux peu profonds

Analyse d'images par réseaux de neurones

- Comment adapter les réseaux de neurones aux images ?
  - Exploitation de la notion de *convolution*



- Paramètres partagés
- Connectivité locale

Moins de paramètres pour des architectures plus profondes

Briques fondamentales

Convolutions, cartes de caractéristiques, champ réceptif (*receptive field*)

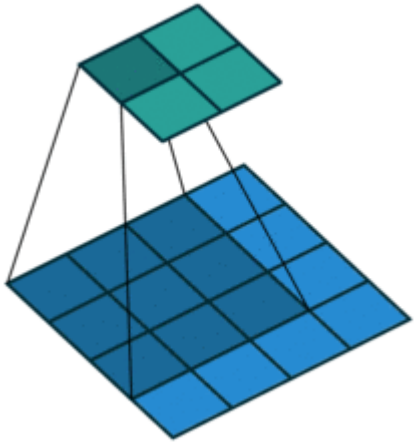
Rappel sur la convolution

$$\begin{pmatrix} 0 & 1 & 2 \\ 0 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix} * \begin{pmatrix} 3 & 3 & 2 & 1 & 0 \\ 0 & 0 & 1 & 3 & 1 \\ 3 & 1 & 2 & 2 & 3 \\ 2 & 0 & 0 & 2 & 2 \\ 2 & 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 12 & 12 & 17 \\ 10 & 17 & 19 \\ 9 & 6 & 14 \end{pmatrix}$$

filtre image sortie

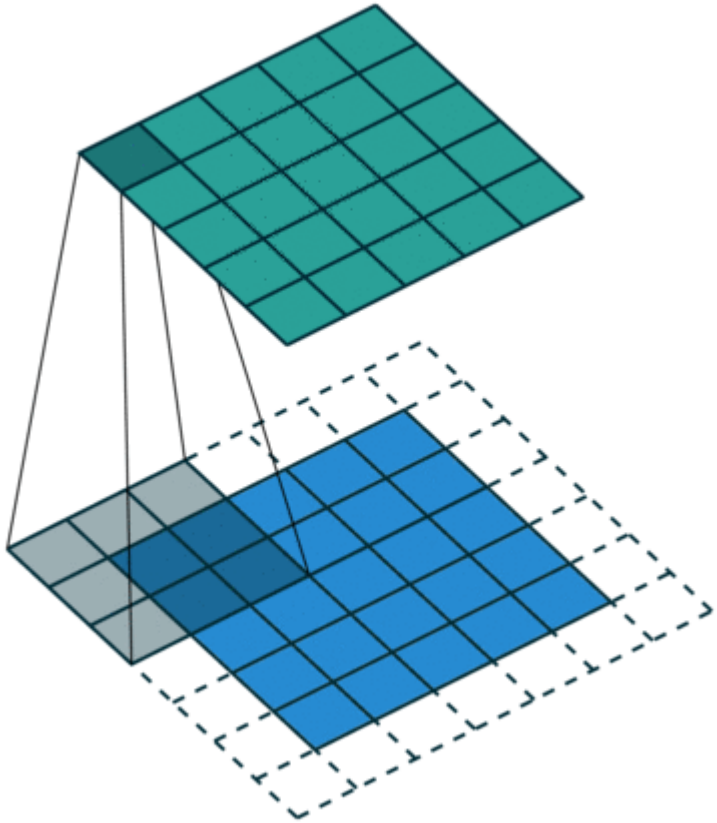
$$G = h * F \qquad G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v] F[i-u, j-v]$$

Remplissage / Padding



Sans padding

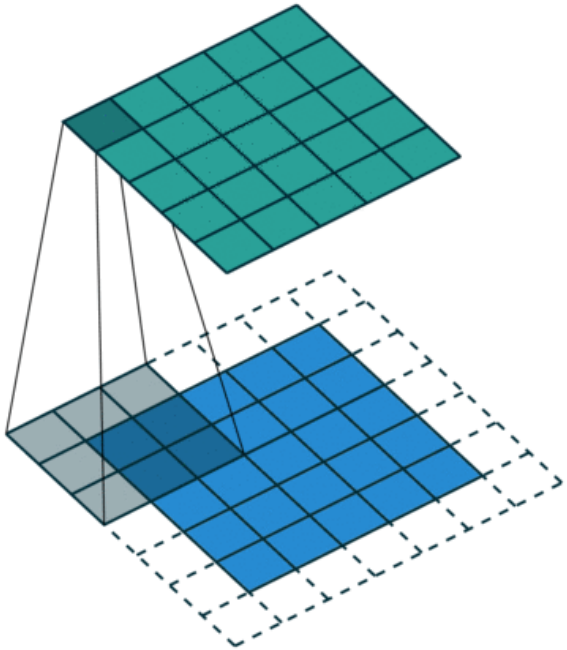
Dimensions spatiales réduites



Avec padding

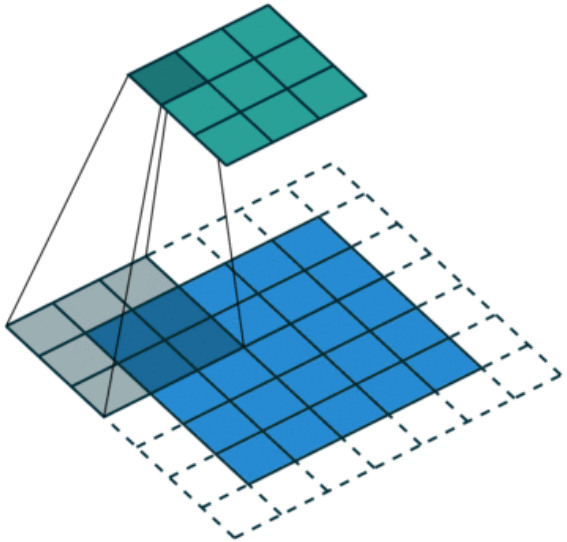
Dimensions spatiales préservées

Pas / Stride



Stride = 1

Dimensions spatiales préservées

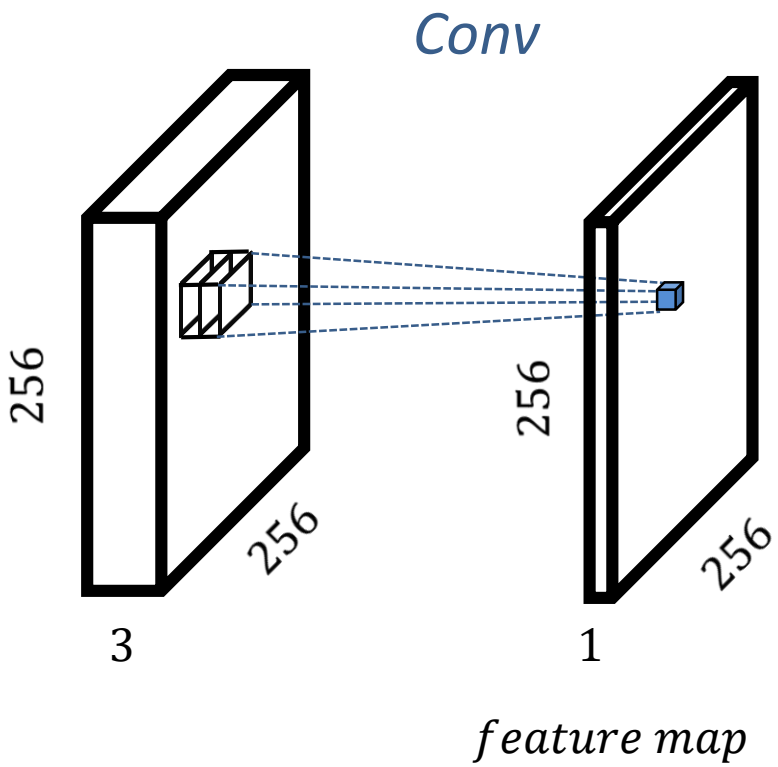


Stride = 2

Dimensions spatiales réduites

Réseaux de neurones convolutionnels

- Paramètres d'apprentissage - *poids de chaque filtre*



- *Ex.* filtre de taille 3×3

$$\#\, param = 3 \times 3 \times 3 + 1$$
$$= 28$$

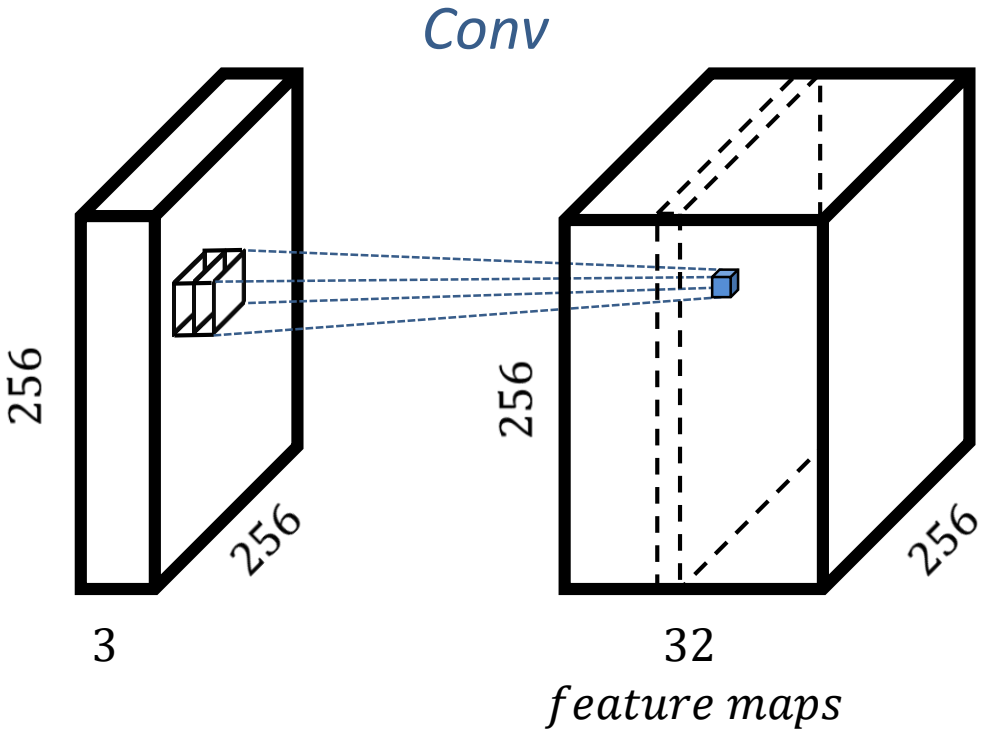
- *Conv*

  filtrage de l'image d'entrée + fonction d'activation

  $feature\ map\ /\ canal$

Réseaux de neurones convolutionnels

- Plusieurs *feature maps* (canaux) par couche

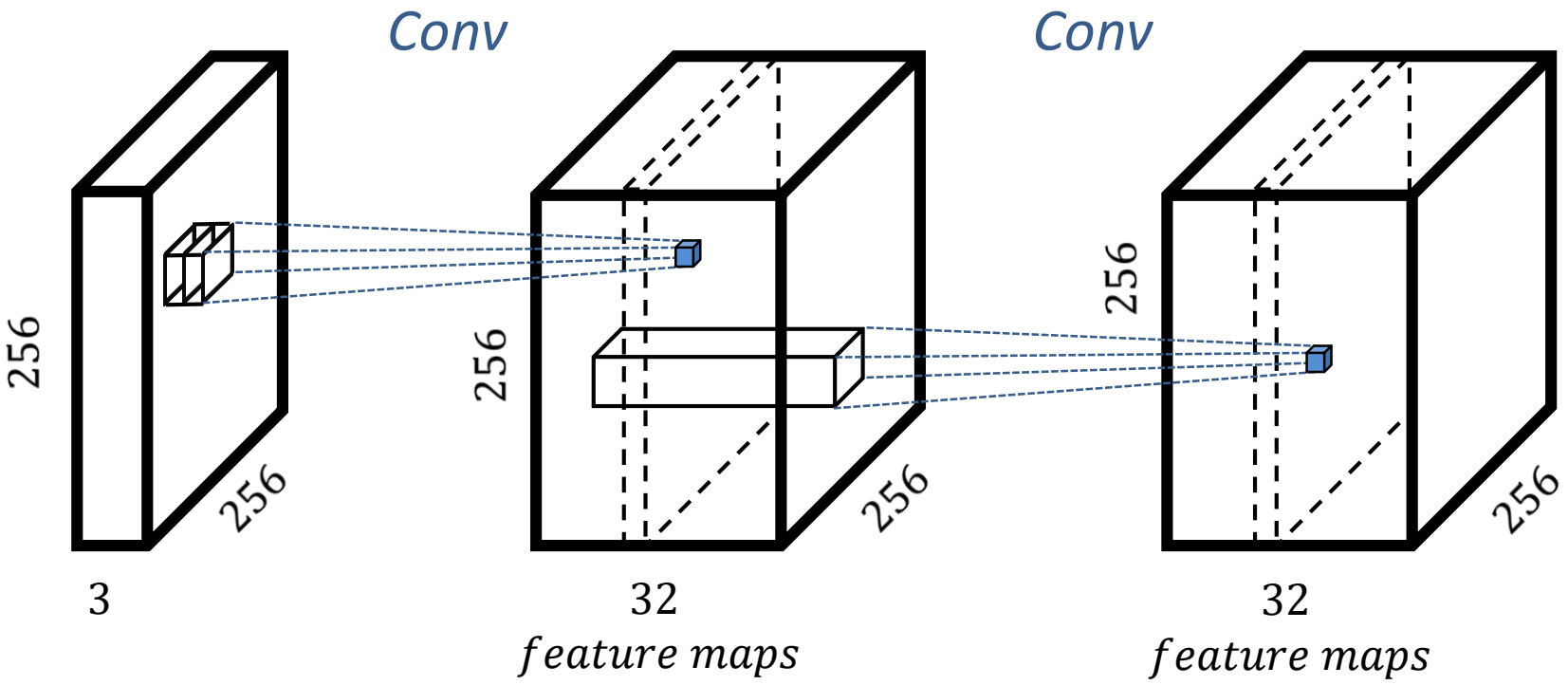


- *Ex.* filtre de taille 3×3

$$\#\, param = 32 \times (3 \times 3 \times 3 + 1) = 896$$

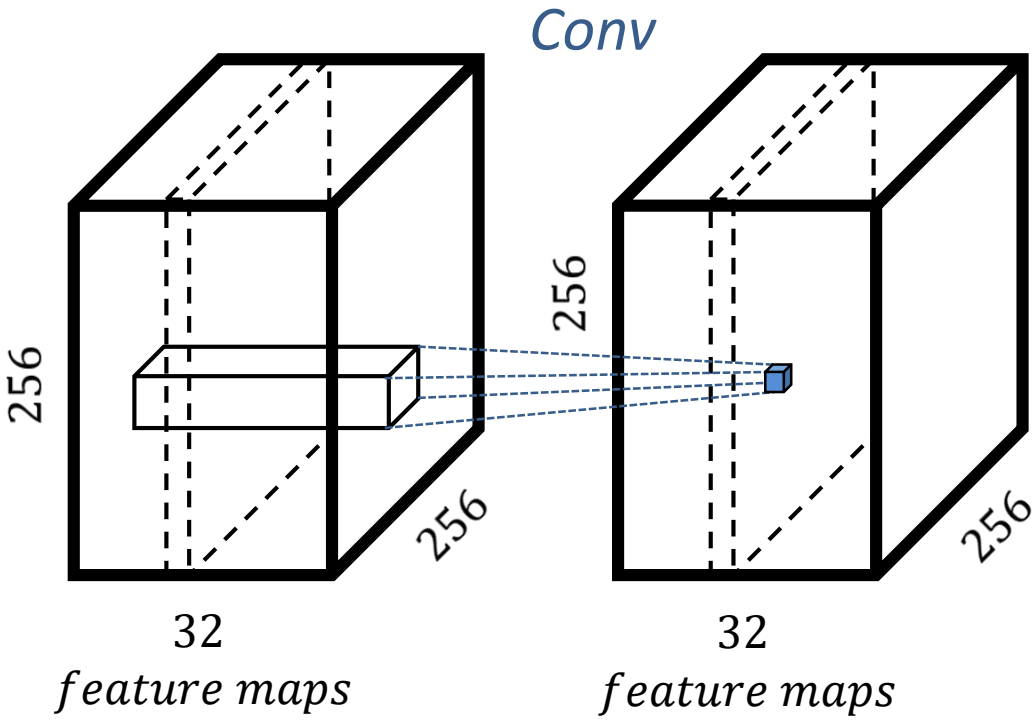
Réseaux de neurones convolutionnels

- Approche multicouches



Réseaux de neurones convolutionnels

- Approche multicouches

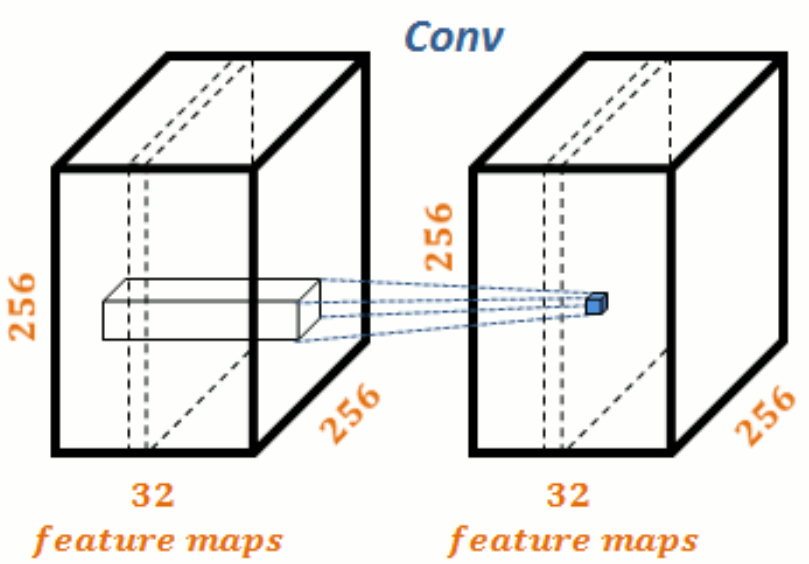


- *Ex.* filtre de taille 3×3

$$\#\, param$$
$$= 32 \times (3 \times 3 \times 32 + 1)$$
$$= 9248$$

Réseaux de neurones convolutionnels

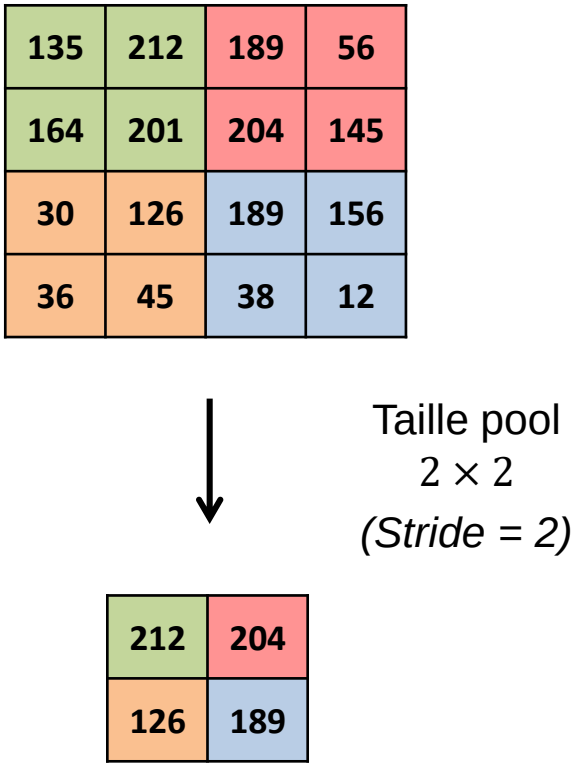
- Approche multicouches
  - Taille du filtre = nombre de canaux d'entrée
  - Nombre de canaux de sortie = nombre de filtres
  - Noyau d'une convolution 2D est un filtre 3D
  - Noyau d'une convolution 3D est un filtre 4D



Pooling

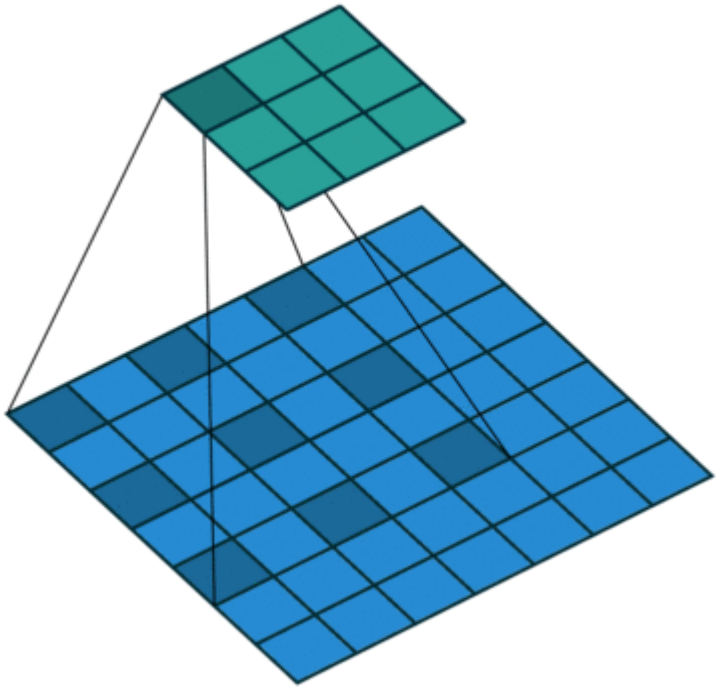
- Réduction de la résolution spatiale des *canaux (feature maps)*
- Diminution de l'empreinte mémoire / coût calculatoire
- Invariance pour de petites translations, rotations et changements d'échelle
- Appliqué individuellement à chaque canal d'entrée

Max pooling



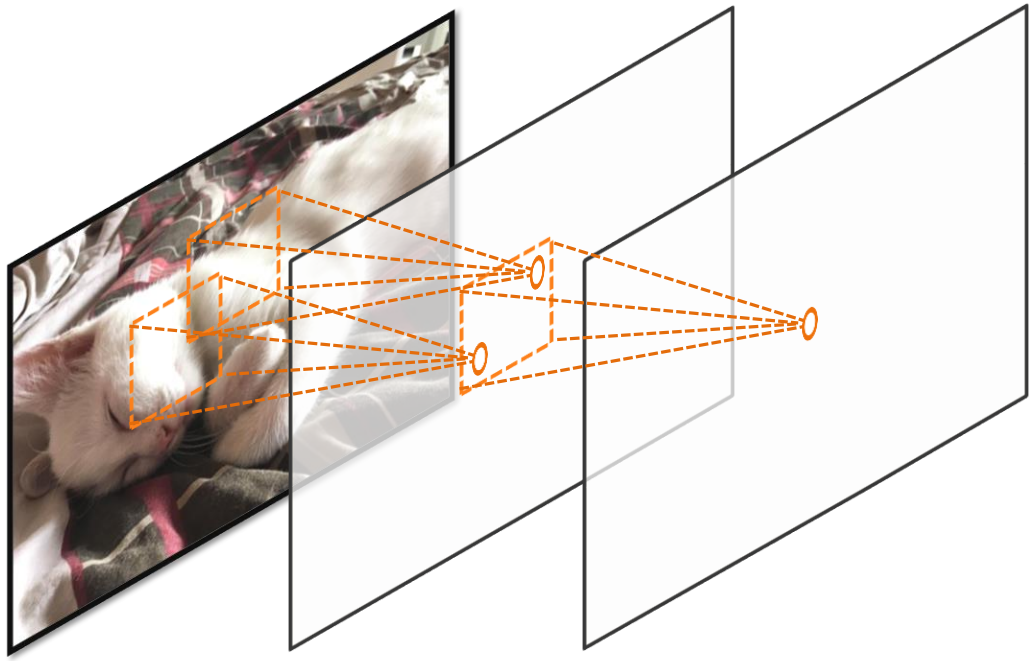
Convolution dilatée (à trous)

- Augmente la taille du filtre en rajoutant des espaces entres les éléments
- Contrôle par le paramètre de dilatation *d*
- Réduit la dimension spatiale des *feature maps* en sortie



Proche de l'opération de pooling mais ici des paramètres doivent être appris

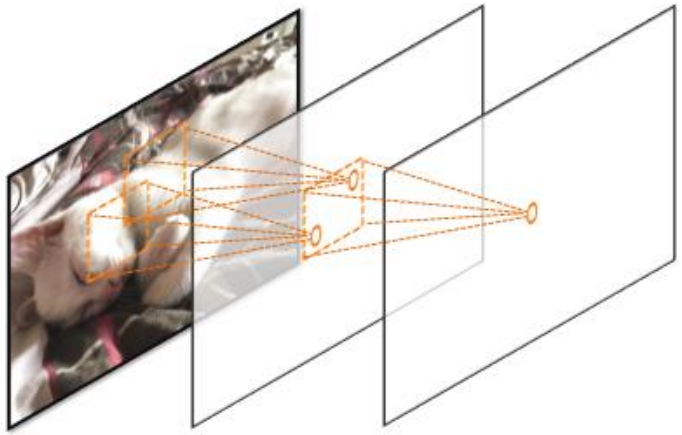
Champ réceptif / Receptive field



Région de l'image d'entrée qui affecte la valeur d'un neurone d'une couche

Champ réceptif / Receptive field

- Un champ réceptif large est nécessaire pour capturer une information de contexte spatial
- Au prix du nombre de paramètres
- Le champ réceptif augmente avec la profondeur d'un réseau

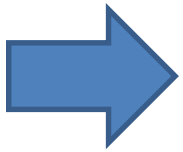


Comment avoir un champ réceptif large sans un nombre trop élevé de paramètres ?

Applications

Classification d'images

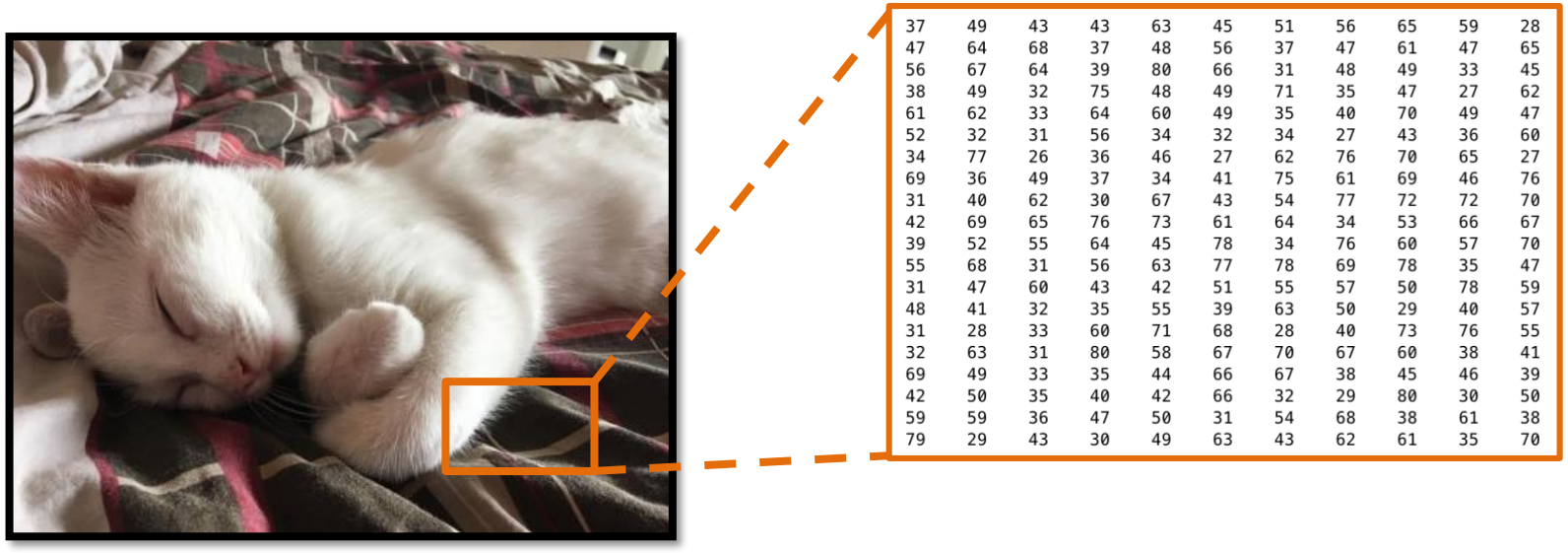
Classification d'images



Chien - 10%
Chat - 85%
Cheval - 5%
Chaise - 0%

Prédire une seule classe (ou une distribution de probabilité sur un ensemble de classes) pour une image donnée

Difficultés



Ce que voit un humain

Ce que voit un ordinateur

D'autres difficultés

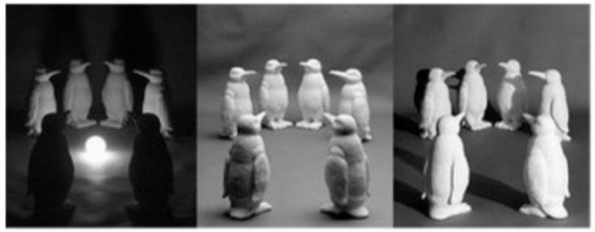
Point de vue d'observation

Variation d'échelle

Déformation

Occlusion

Conditions d'illumination



Texture

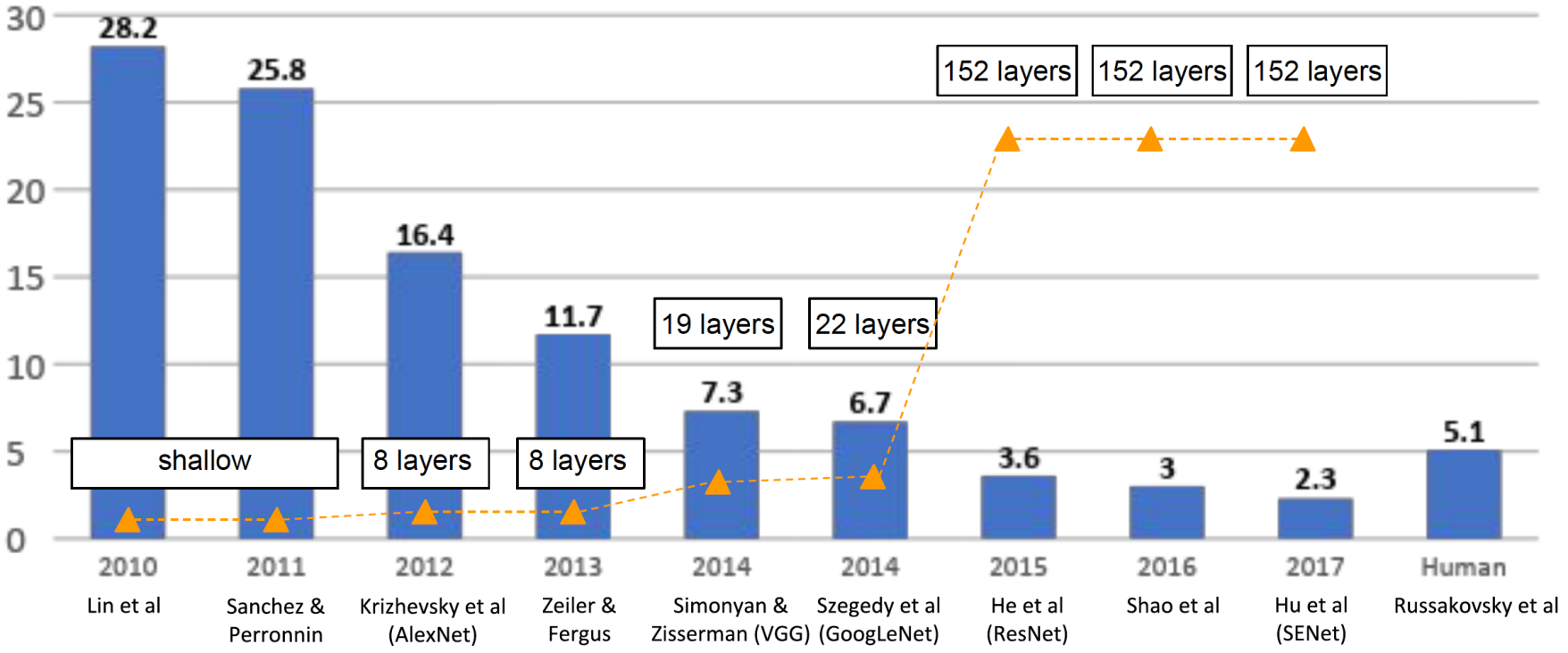
Variation intra-class

Simple pour un humain, qu'en est-il pour un ordinateur ?

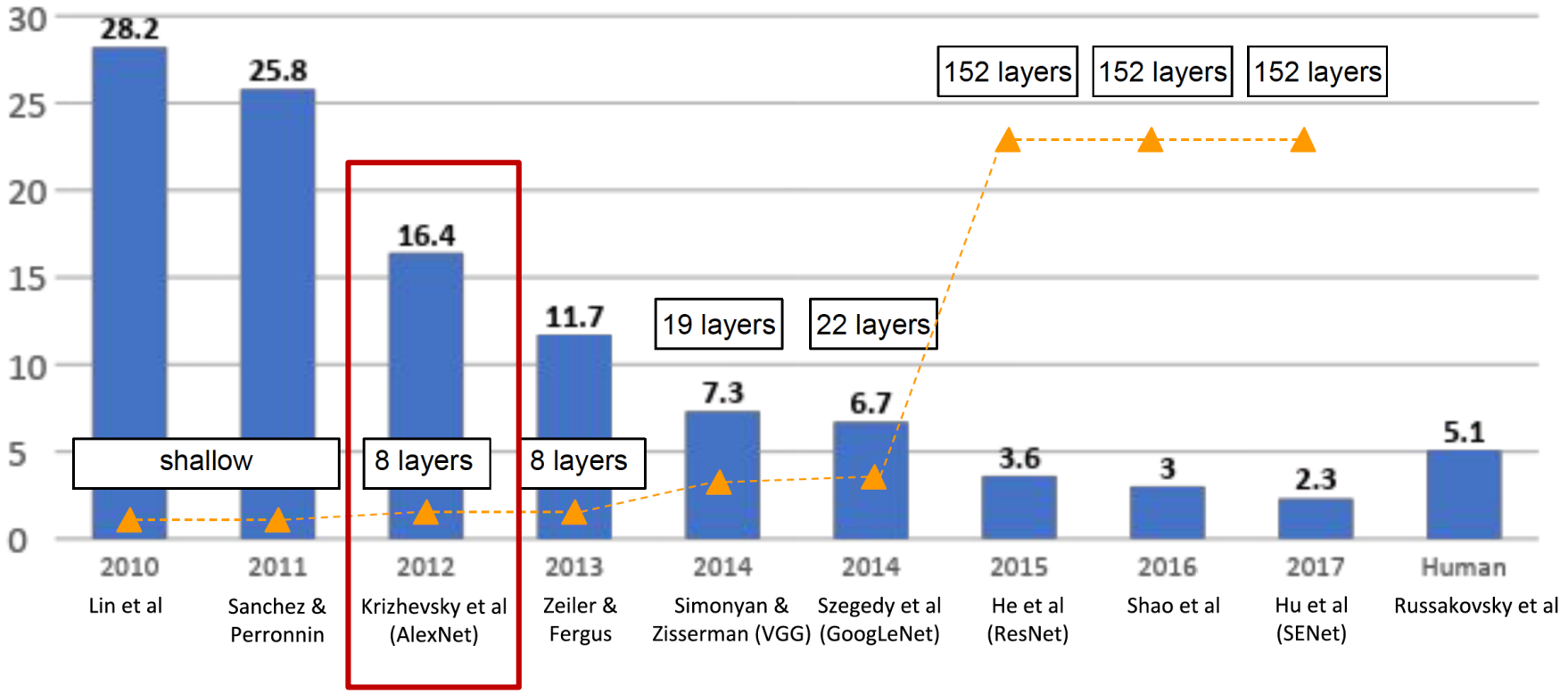
- Challenge en classification d'images (2010 → 2017)
- 1 000 classes d'objet à reconnaitre
- 1 431 167 images

ImageNet

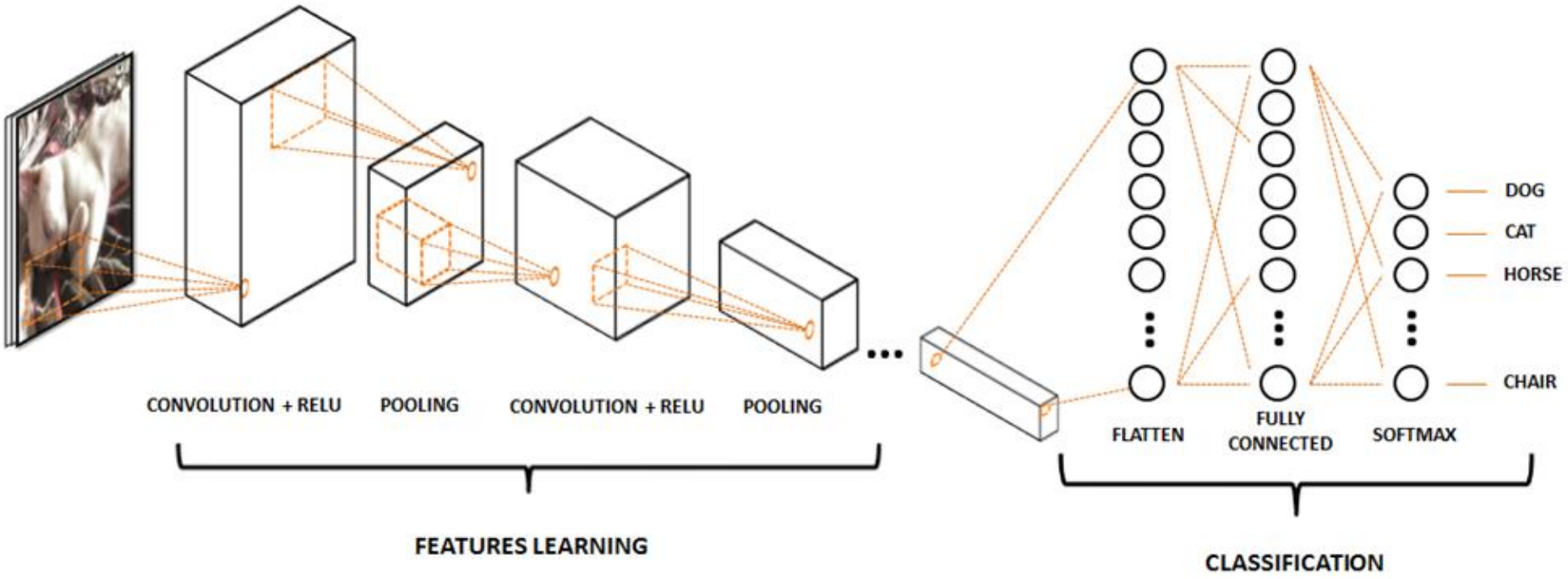
- Classement annuel



AlexNet [Krizhevsky, NIPS, 2012]



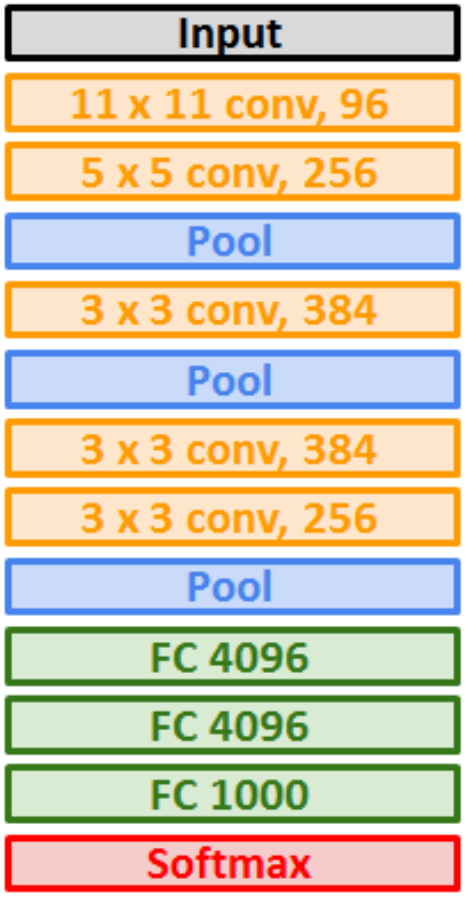
AlexNet [Krizhevsky, NIPS, 2012]



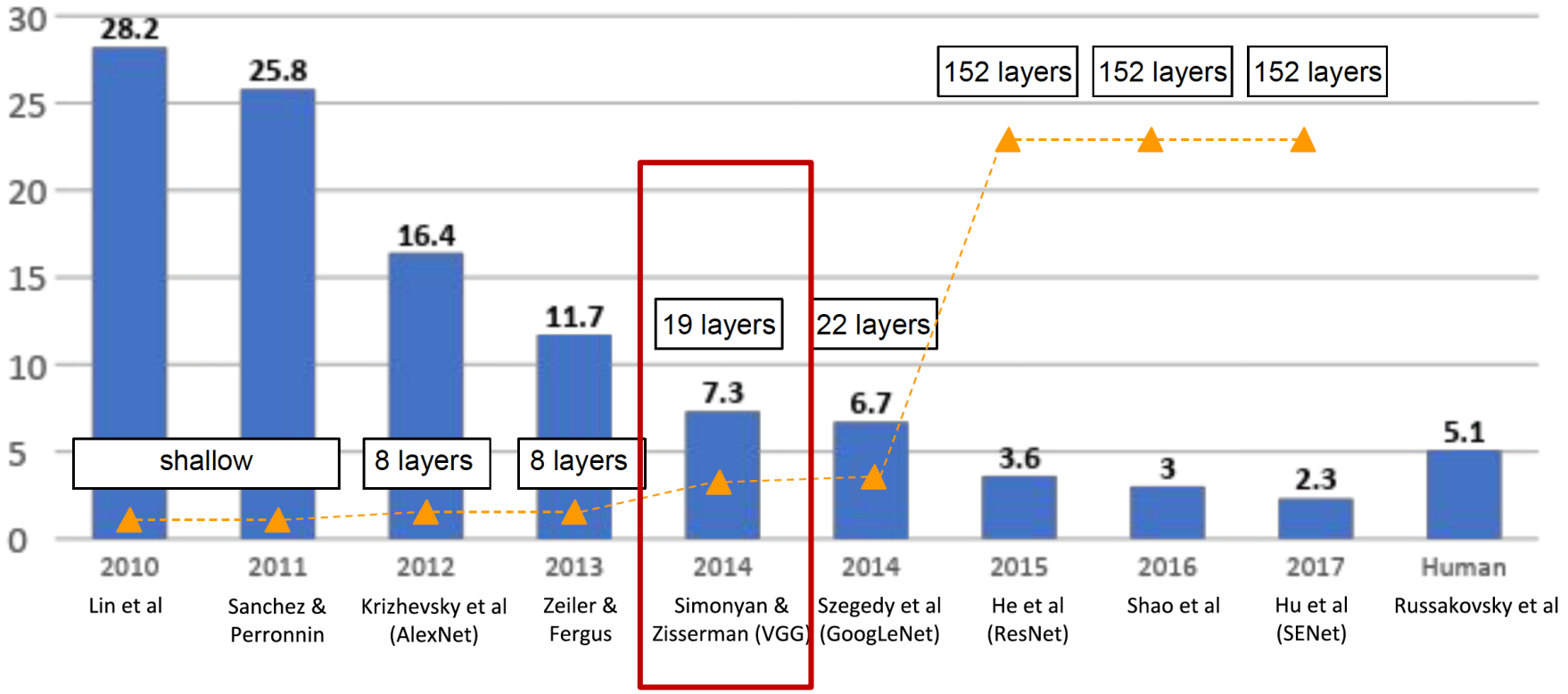
AlexNet [Krizhevsky, NIPS, 2012]

- Premier modèle à bien performer sur la base de données ImageNet
- Exploite des techniques encore utilisées (ReLU, augmentation de données, dropout)
- Utilise des GPU pour l'entrainement

Responsable de la révolution de l'apprentissage profond en vision par ordinateur



VGG [Simonyan and Zisserman, arxiv, 2014]



VGG [Simonyan and Zisserman, arxiv, 2014]

- Architecture plus simple

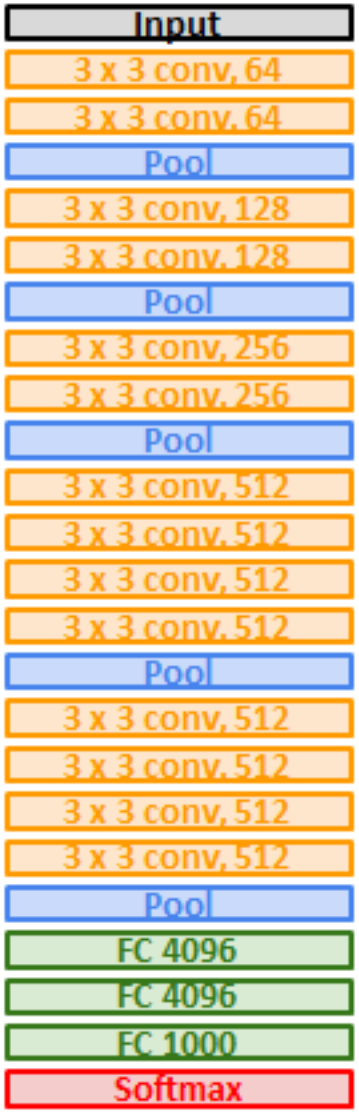
  Convolutions 3×3, ReLU et max pooling 2×2

- Réseau plus profond

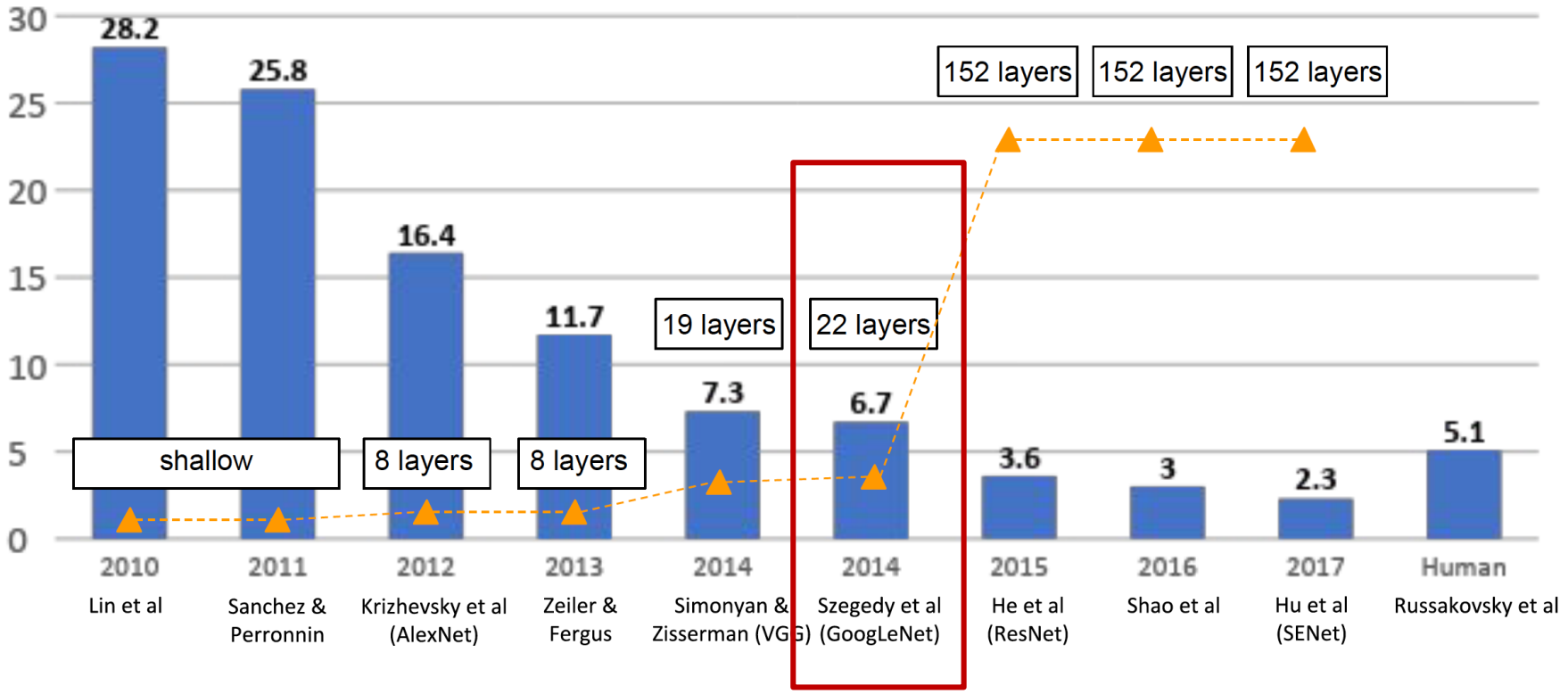
  19 couches (8 couches pour AlexNet)

- Idée clé:

Mettre en cascade 2 convolutions 3×3 produit le même champ réceptif qu'un convolution 5×5 mais avec moins de paramètres

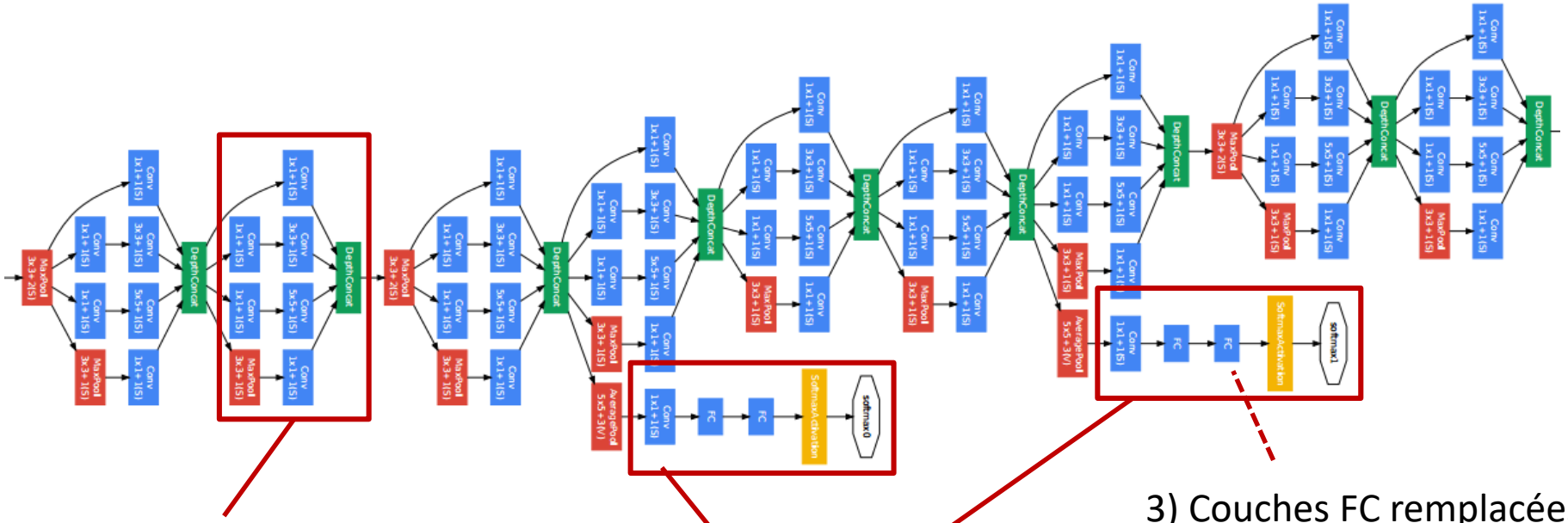


GoogLeNet (Inception v1) [Szegedy, CVPR, 2015]



GoogLeNet (Inception v1) [Szegedy, CVPR, 2015]

- Réseau entièrement repensé pour être très profond



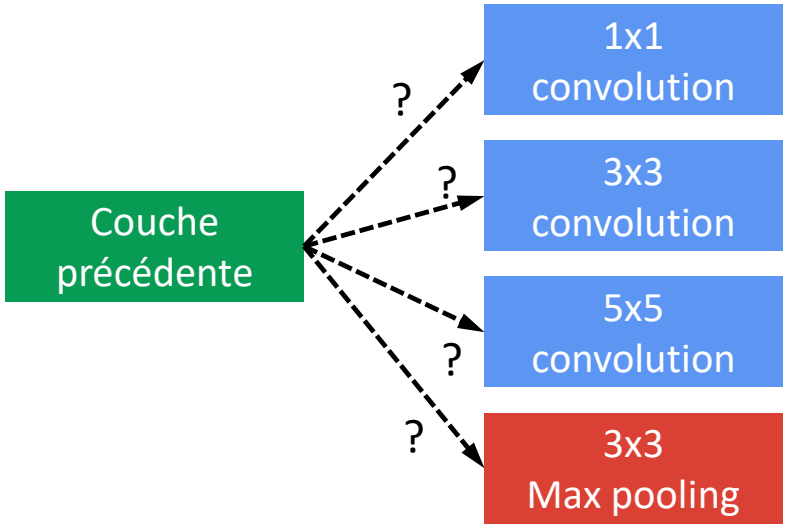
1) Blocs répétitifs
Inception module

2) Fonctions de pertes intermédiaires afin d'injecter des gradients dans les couches intermédiaires

3) Couches FC remplacées par du average pooling (moins de paramètres)

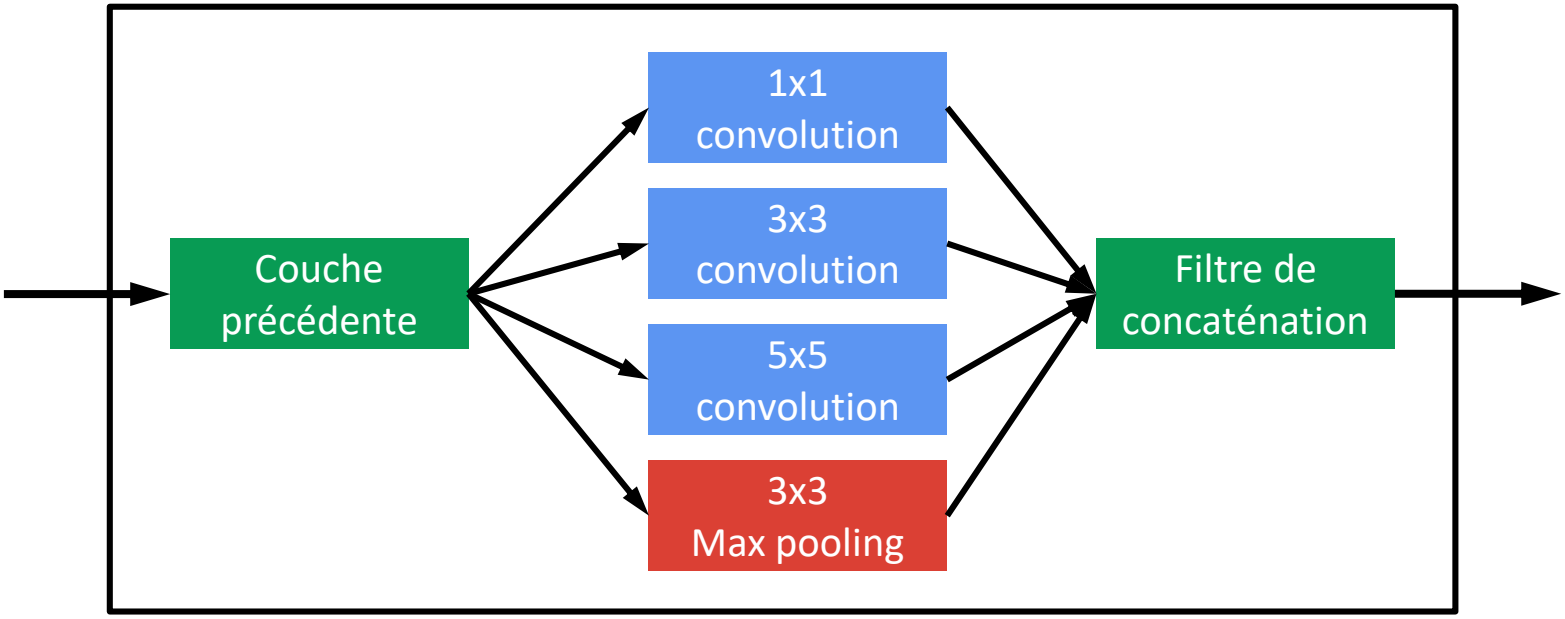
Inception module

- Choix pour chaque couche
  - Convolution ou pooling ?
  - Si convolution, quelle taille de filtre ?



Inception module – Idée clé

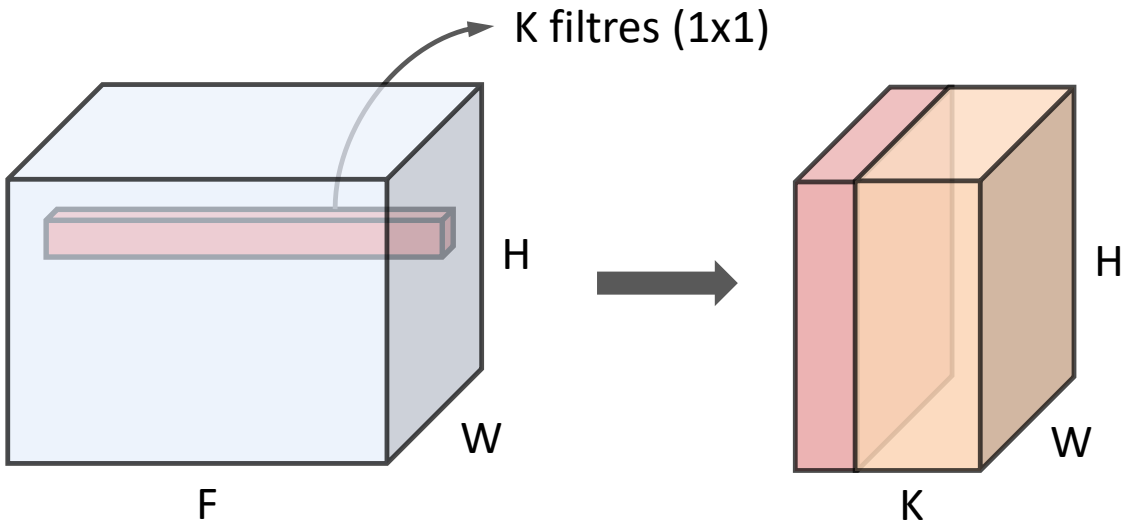
- Calculer toutes les sorties en parallèle
- Concaténation des résultats
- Laisser l'apprentissage choisir !



Problème: cela produit trop de sorties et de paramètres

Inception module – Idée clé

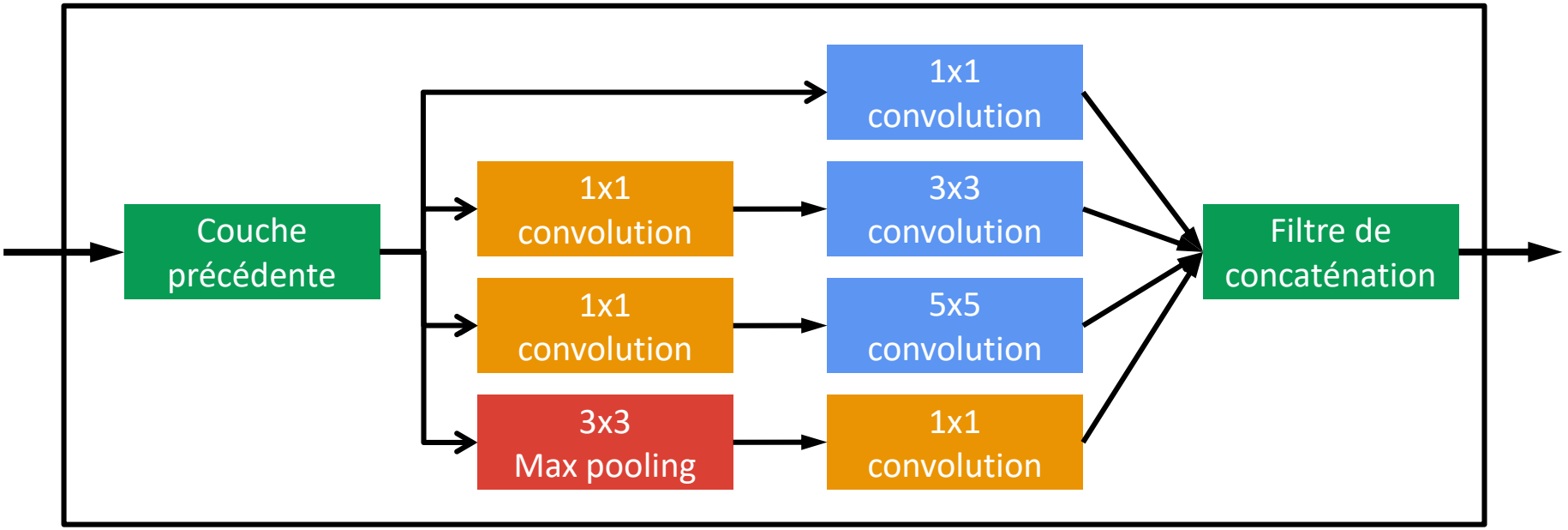
- Convolution 1×1



Réduction de la dimension pour K<F

Agit comme une fonction de « feature pooling » que l'on peut apprendre

Inception module – Idée clé



Réduction des dimensions via l'utilisation de couches d'étranglement composées de convolutions 1×1

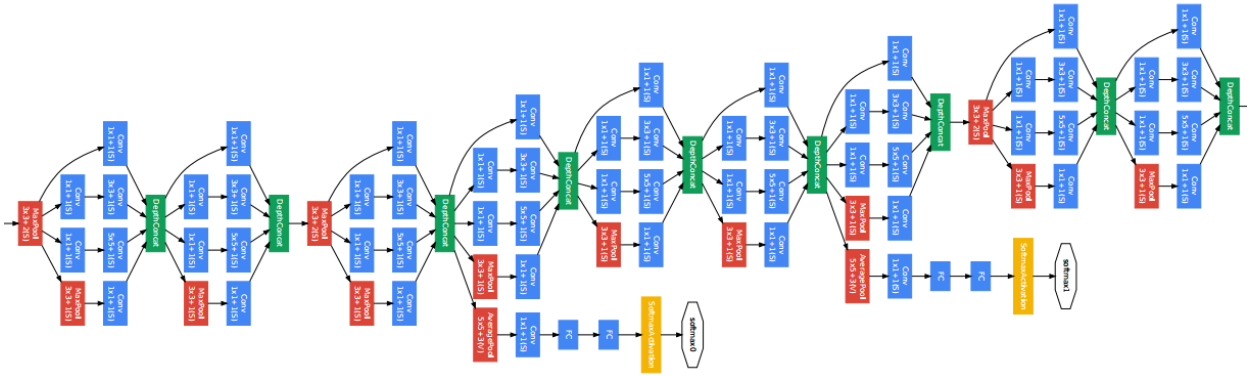
Efficacité

AlexNet

8 couches
param ~ 62M

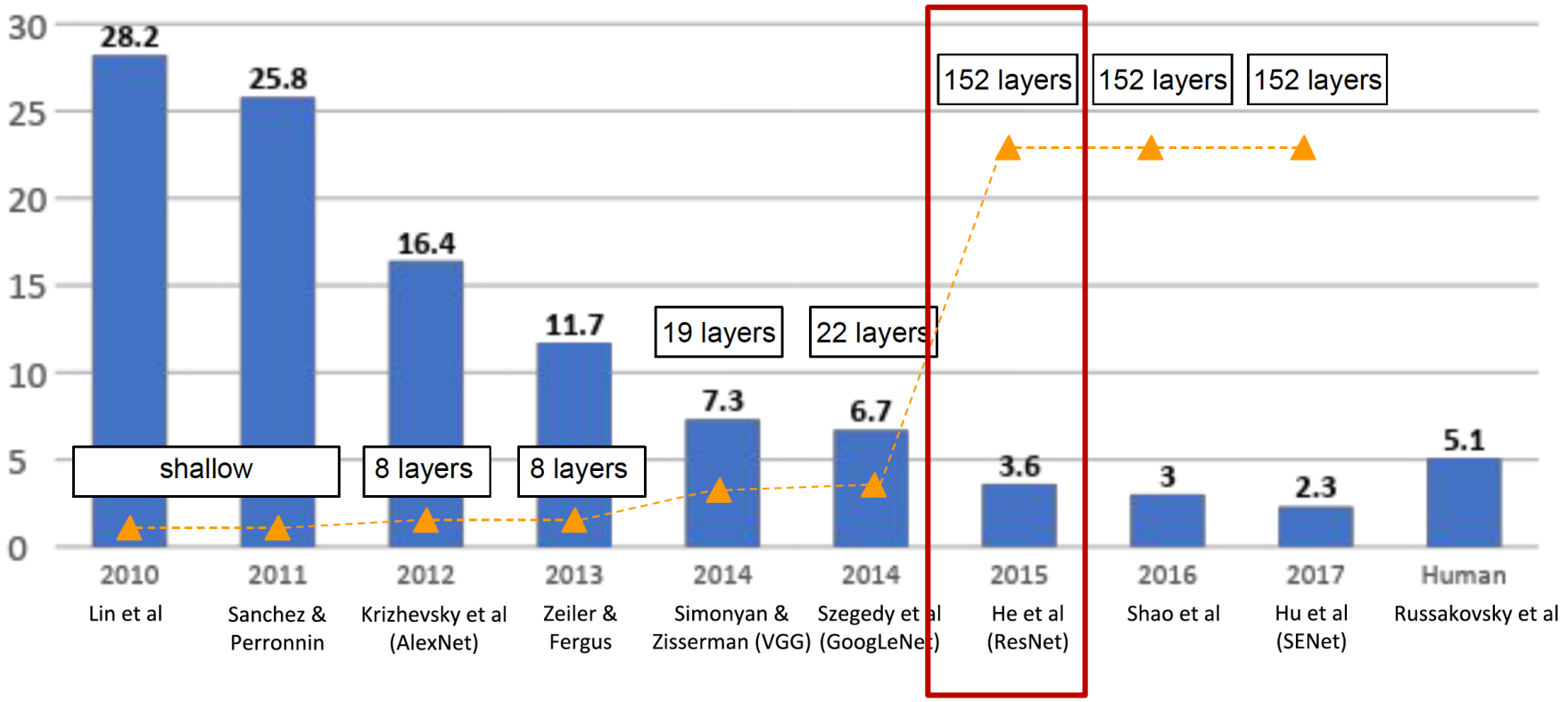
GoogLeNet

22 couches
param ~ 5M



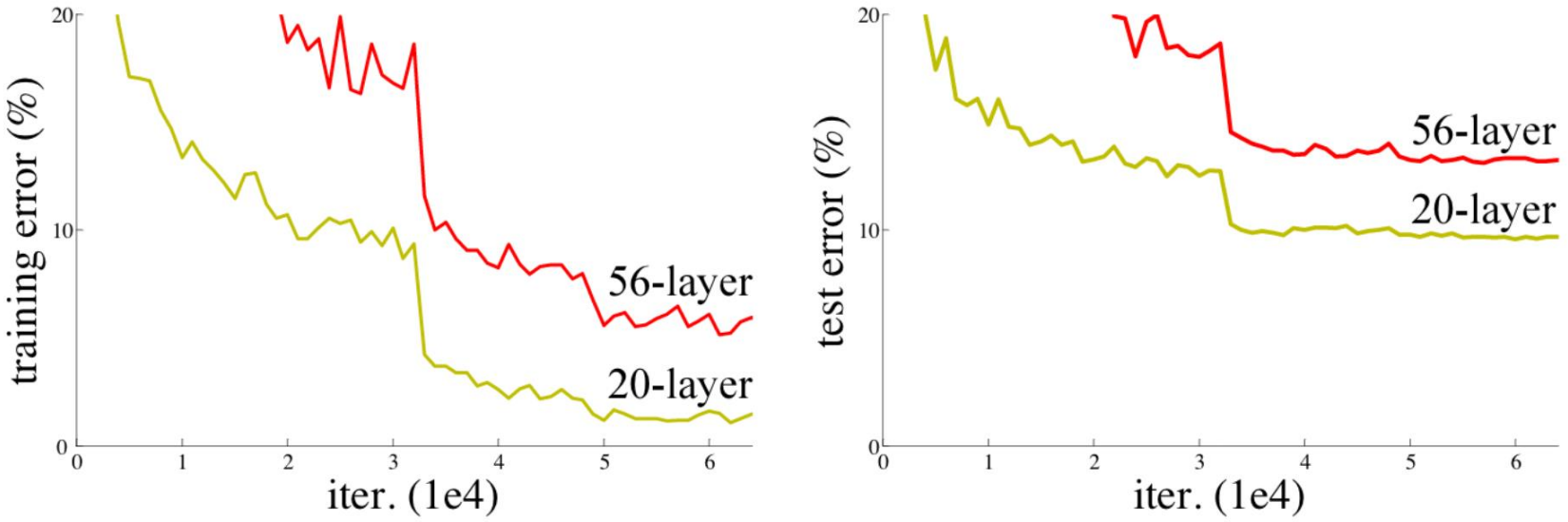
GoogLeNet possède 12x moins de paramètres qu'AlexNet !

ResNet [He, CVPR, 2016]



ResNet [He, CVPR, 2016]

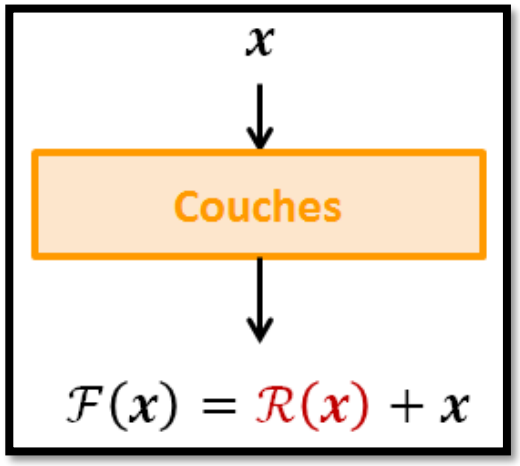
- Que se passe t'il pour un réseau encore plus profond ?



- Erreurs d'entrainement et de test plus grandes !

Problème d'optimisation (*vanishing gradient*)

ResNet [He, CVPR, 2016]

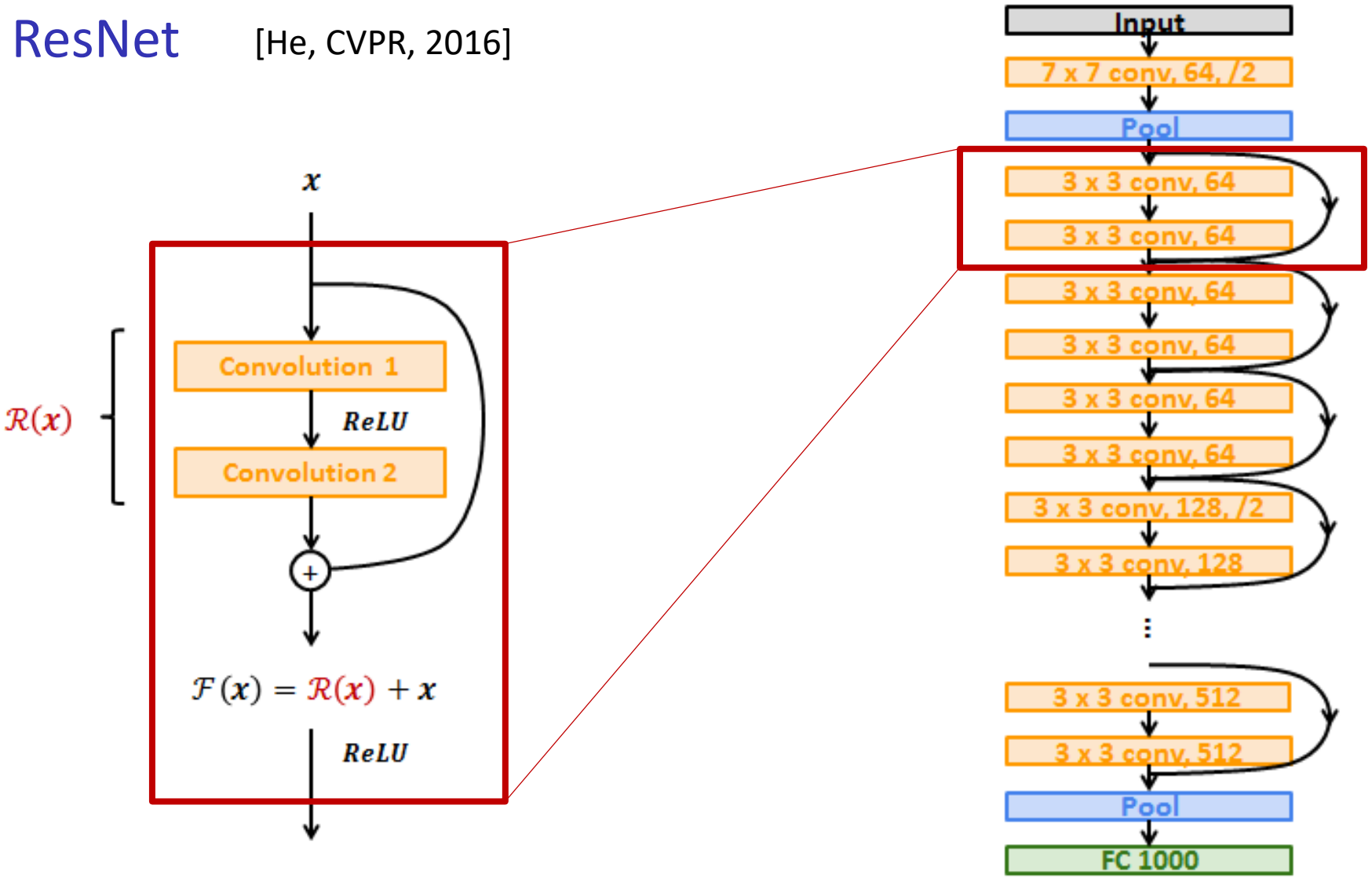


- Idée clé

Estimer le résidu plutôt que la transformation

- Avantages
  - Modélisation de moins d'information, potentiellement plus simple à apprendre
  - Connections résiduelles préservent le flux de gradient lors de la propagation arrière
  - Conception d'architectures très profondes ($>$ 100 couches)

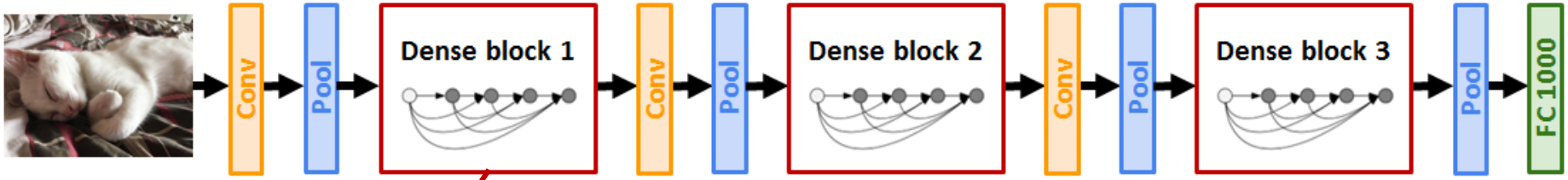
ResNet [He, CVPR, 2016]

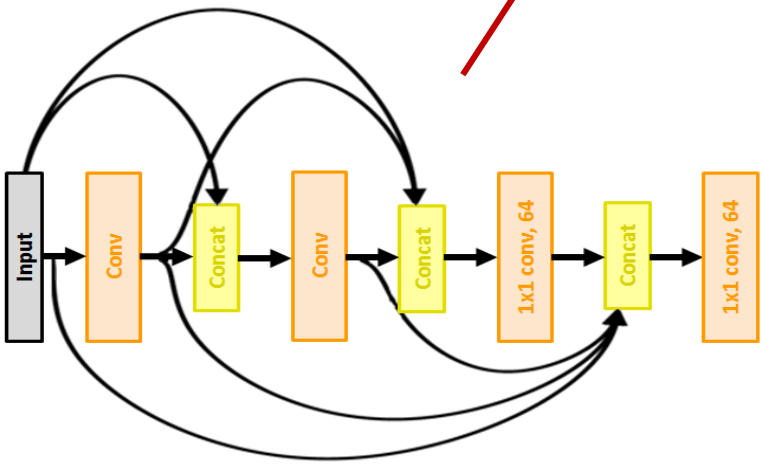


DenseNet (Densely connected) [Huang, CVPR, 2017]

- Idée clé

Les *features* calculées dans une couche sont concaténées avec les entrées de toutes les autres couches d'un bloc





- Utilisation efficace de *features* multi-échelles
- Propagation du gradient au travers de chaque couche lors de la propagation arrière

Applications

Segmentation sémantique

Segmentation sémantique d'images

- Prédire la bonne classe pour chaque pixel d'une image

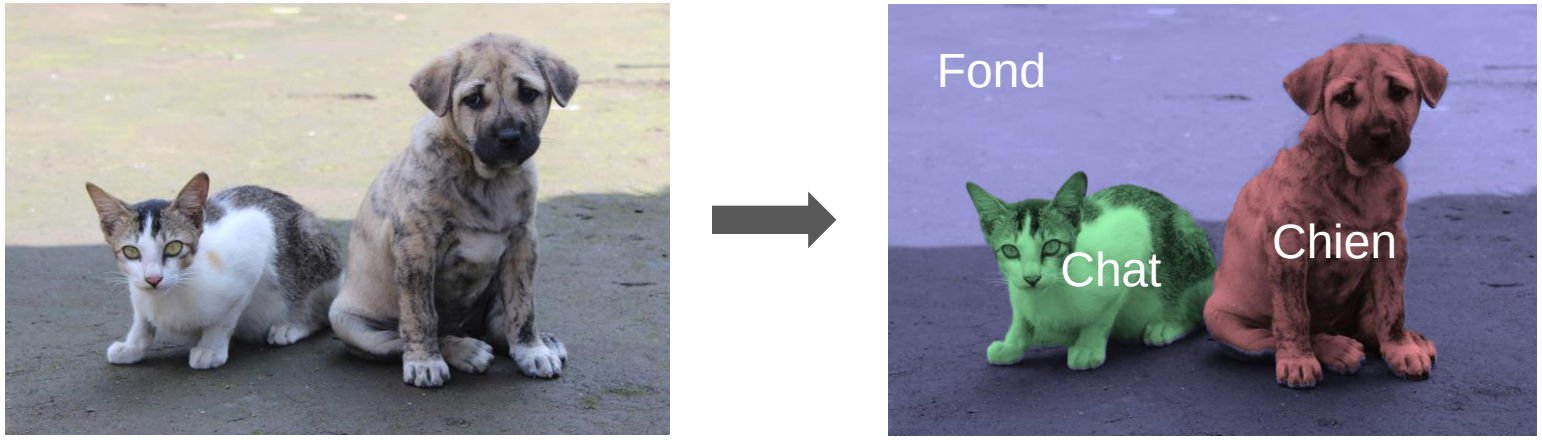
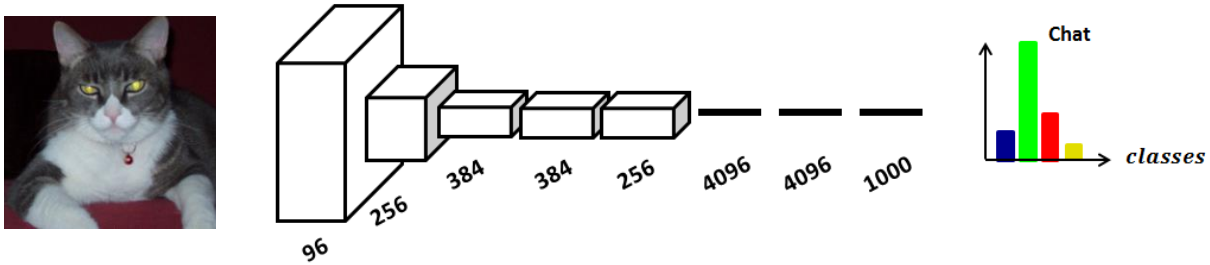


Image d'entrée *Segmentation*

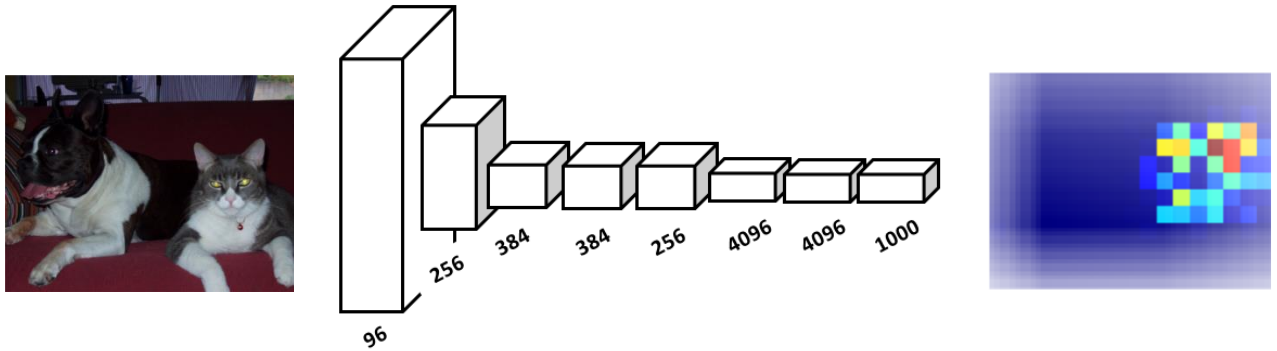
Peut-être vu comme un problème de classification dense et structuré

Fully-CNN: de la classification à la segmentation

Classification standard CNN



Fully-CNN

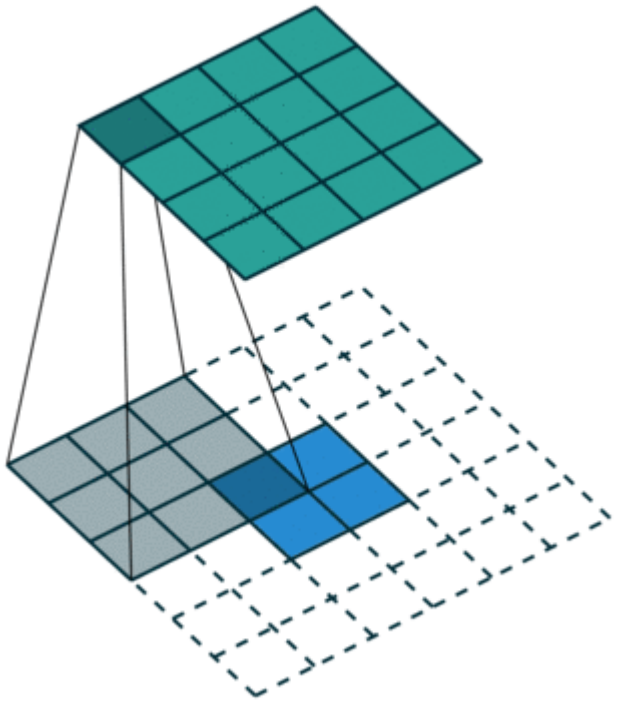


- Génération de cartes de segmentation très grossières

Ajout d'opérations de sur-échantillonnage à la fin du réseau

Long et al. "Fully convolutional networks for semantic segmentation" ICCV, 2015

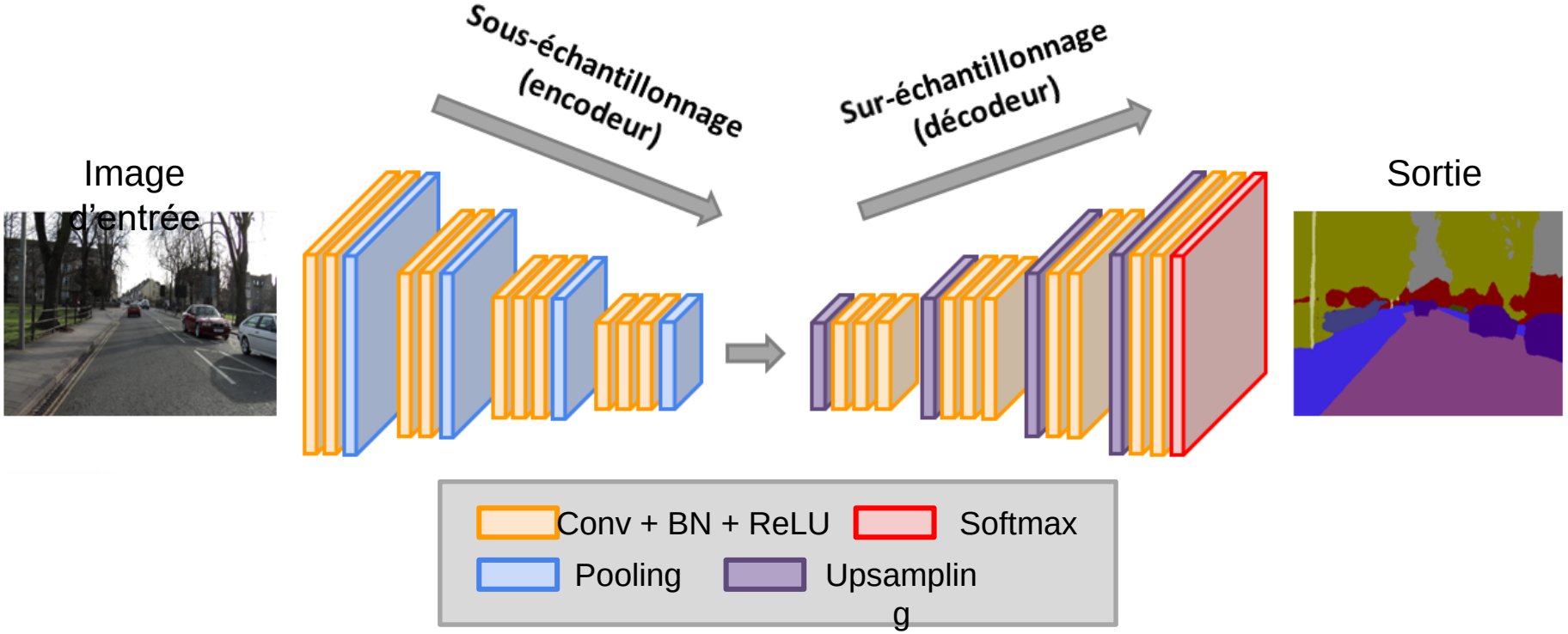
Opération de sur-échantillonnage



Padding 2, stride = 1

Dimensions spatiales augmentées

Architectures de type encodeur / decodeur

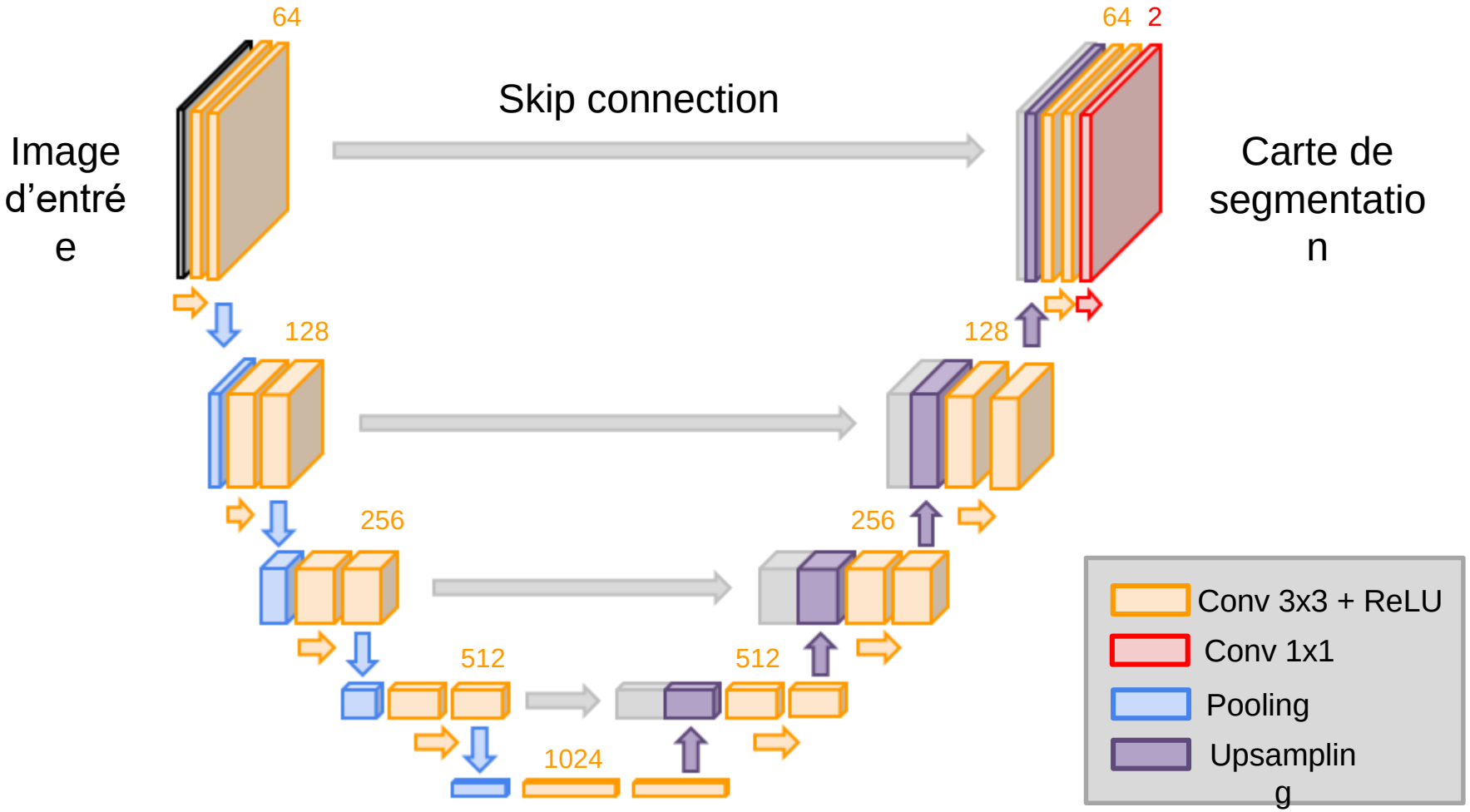


- Résolution spatiale perdue durant le sous-échantillonnage

Ajout de « skip connections » entre l'encodeur et le décodeur

Badrinarayanan et al. "Segnet: a deep convolutional encoder-decoder architecture for image segmentation " PAMI, 2017

U-Net



Ronneberger et al. "U-net: Convolutional networks for biomedical image segmentation" MICCAI, 2015

nnU-Net

- Méthode basée sur l'architecture U-Net
  - Mise en forme automatique de la base de données
  - Planification des patchs / topologie de réseaux / batchs
  - Stratégie efficace de généralisation des réseaux
    - ✓ Augmentation de données en entrainement et en inférence
    - ✓ Nombre d'itération maximum
    - ✓ Décroissance progressive du taux d'apprentissage
    - ✓ Fusion de résultats de plusieurs U-Net

nnU-Net

► Mise en forme automatique de la base de données

- Rééchantillonnage de l'ensemble de la base de données suivant la résolution médiane

  ✓ Permet de travailler sur des images homogènes vis-à-vis des dimensions physiques

- Normalisation des intensités

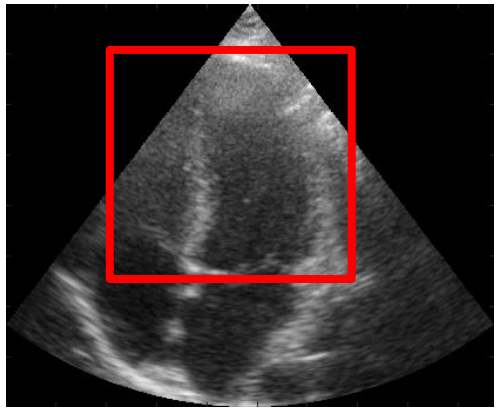
  ✓ Tiens compte uniquement des pixels non nulles.

  ✓

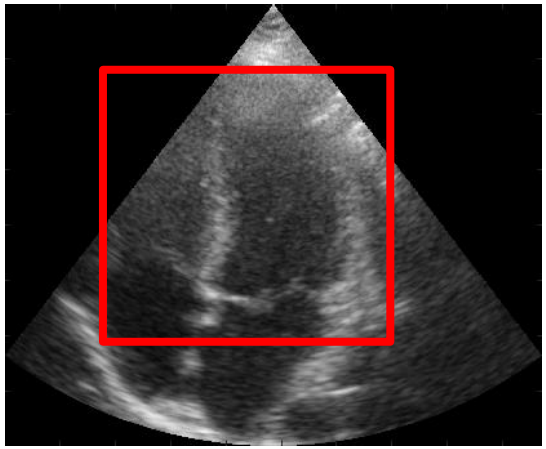
nnU-Net

- Planification des patches / topologies de réseaux / mini-batch
  - Dimensionnement en tenant compte de la capacité du GPU
  - Données d'entrées: patches de taille maximale
    - ✓ permet d'éviter de redimensionner les images

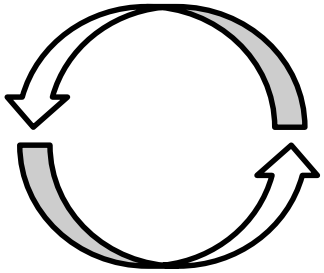
Patch à mettre en entrée d'un réseau U-Net



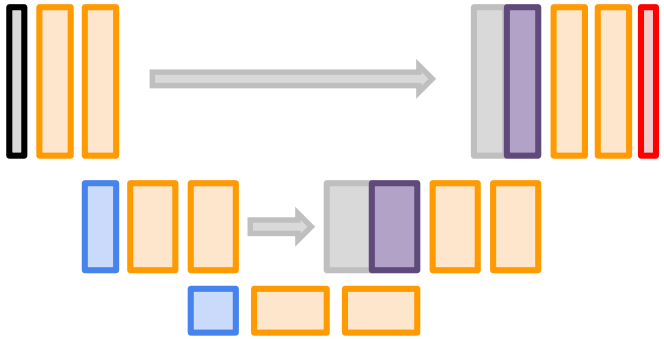
Patches



Dimensions max.



Topologie de réseaux



Profondeur max.

Jusqu'à feature map 2x2

Mini-batch

Taille min. = 2

nnU-Net

- Stratégie de généralisation des réseaux

  - Sélection aléatoire de patch dans chaque image lors de la création des batch

  - 1000 epoch avec 250 itérations par epoch
    - ✓ On ne verra pas toute la bd par epoch => bagging !

  - Décroissance progressive du taux d'apprentissage au cours des epochs
    - ✓Convergence progressive vers un optimum

nnU-Net

► Stratégie de généralisation des réseaux

- Augmentation de données lors de la génération de batch
  - ✓ Rotation, changement d'échelle, effet miroir
  - ✓ Changement d'intensité, floutage gaussien, contraste, luminosité

- Augmentation de données en inférence
  - ✓ Segmentation d'une image et de ses versions miroir
  - ✓ Moyennage des résultats obtenus après effet miroir inverse pour filtrer les zones incertaines

nnU-Net

► Pour la segmentation d'images 3D

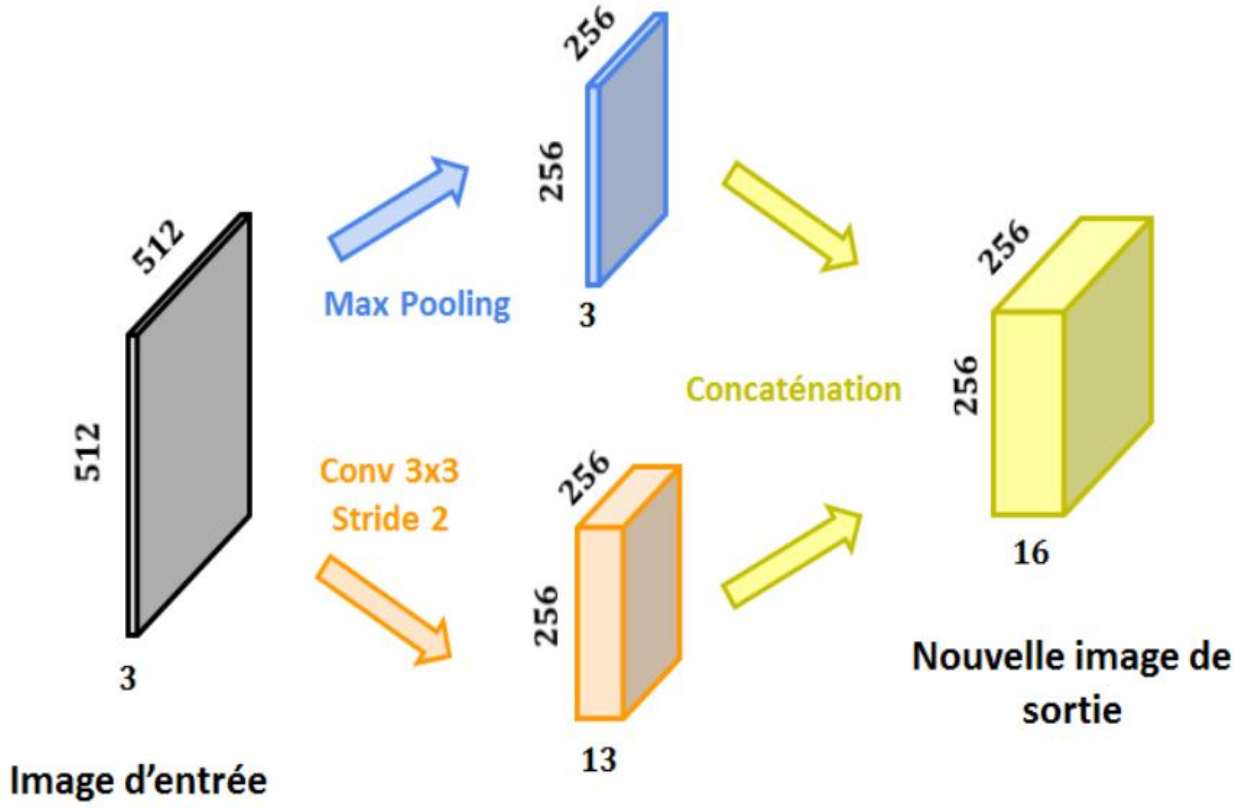
- Apprentissages indépendants de plusieurs réseaux
  - ✓ U-Net 2D / patches 2D
  - ✓ U-Net 3D / patches 3D
  - ✓ U-Net 3D / volumes entiers sous résolus
- Approche de type Boosting
  - ✓ Moyennage des résultats de deux architectures sur la base de données de validation au cours de l'apprentissage
  - ✓ La meilleure combinaison de réseaux est utilisée en inférence

nnU-Net

- Segmentation en inférence
  - Utilisation d'une fenêtre glissante
    - ✓Patches avec un chevauchement de la moitié de la taille du patch
    - ✓Moyennage des résultats obtenus

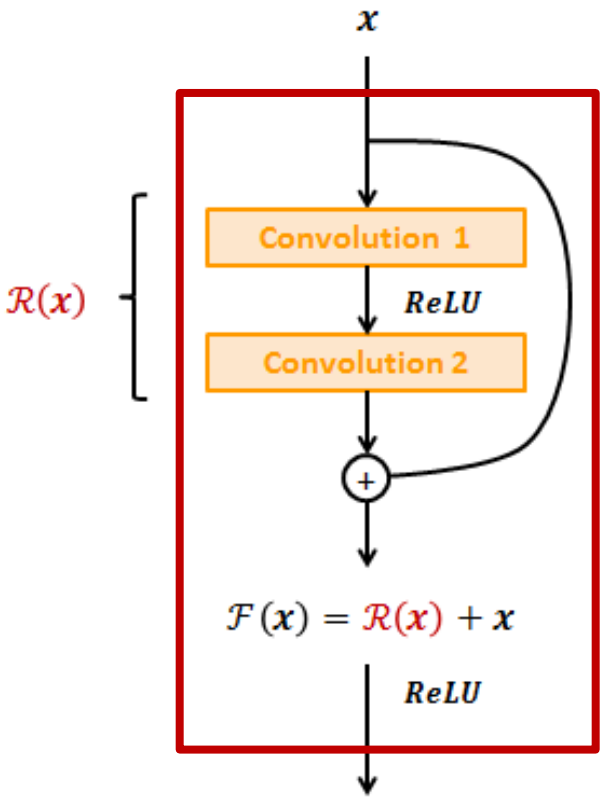
ENet

- Modélisation de l'image d'entrée

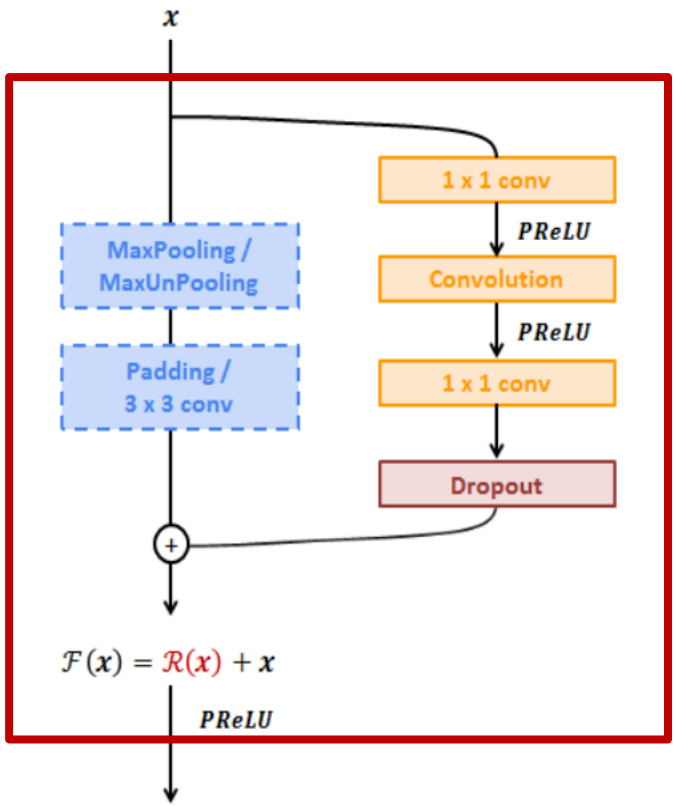


ENet

- Architecture basée sur ResNet



Bloc ResNet



Bloc ENet

Projection / réduction de dimensionnalité

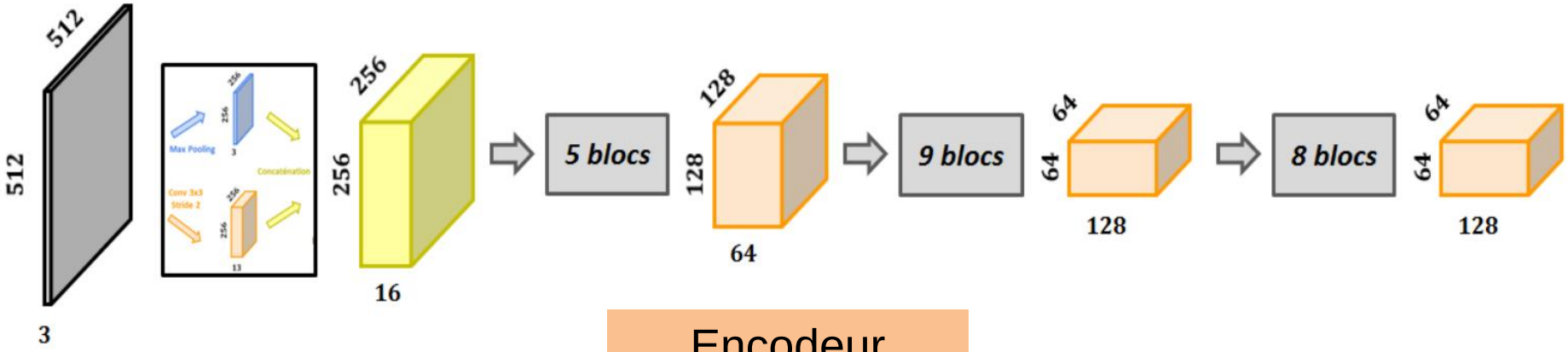
Création d'information

Expansion de l'info au travers de canaux

Régularisation

ENet

- Architecture asymétrique



ENet

► Performances

- Qualité de segmentation équivalente voir meilleure que l'état de l'art en apprentissage profond
- # paramètres: 0.37 M
- Taille du réseau < 6 MB
- Temps d'exécution (NVIDIA TitanX)

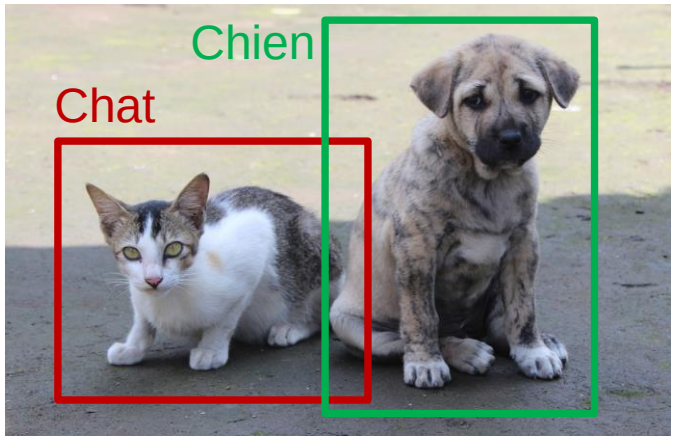
640x360 px	7 ms
1280x720 px	21 ms
1920x1080 px	46 ms

Applications

Détection d'objets

Détection d'objets

Image d'entrée

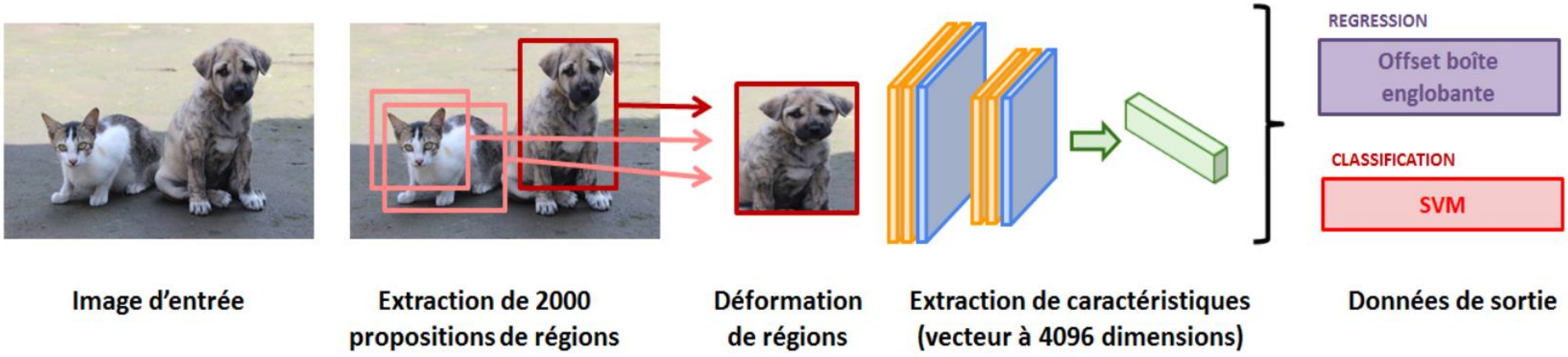


Classes détectées

Trouver les objets/classes présents dans une image ainsi que leur localisation

R-CNN (Region-CNN)

- Phase d'entrainement

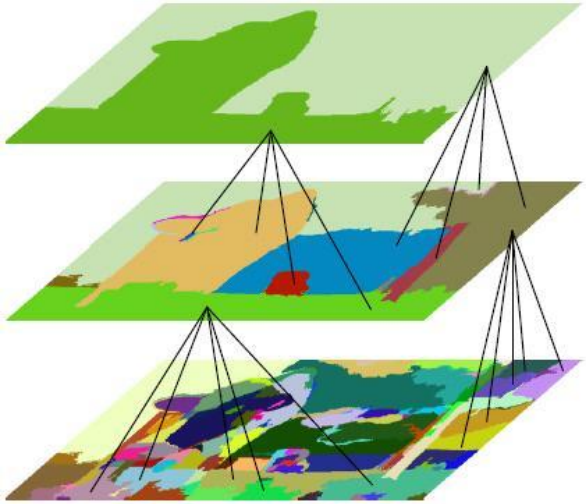


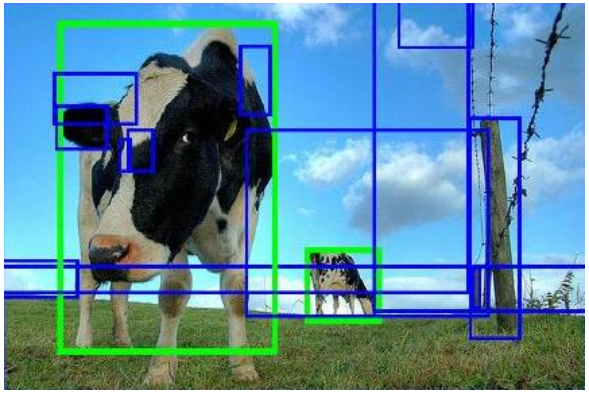
Plusieurs étapes d'entrainement (CNN, SVM, régression boites englobantes)

Girshick et al. "Rich Feature hierarchies for accurate object detection and semantic segmentation" CVPR, 2014

R-CNN

- Extraction de régions



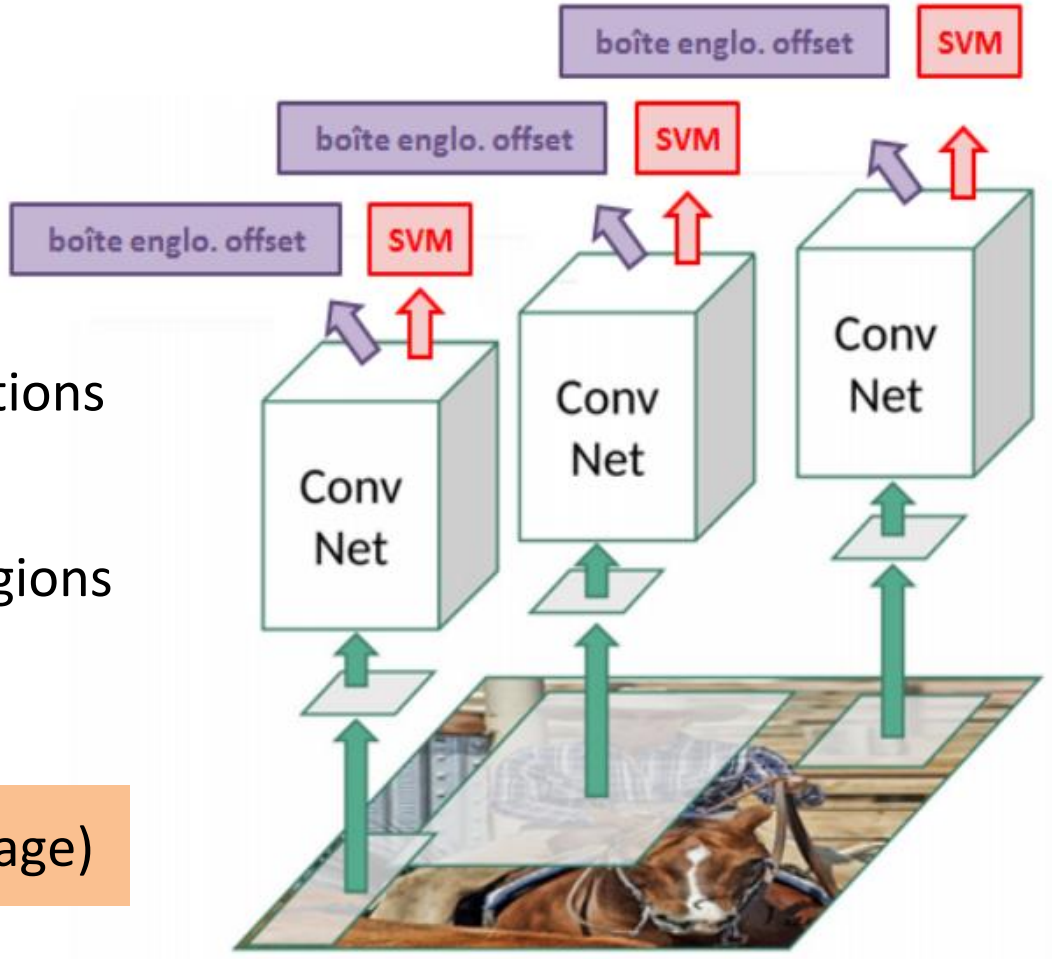


- Méthode classique par graphe

Génération non négligeable de mauvais candidats

R-CNN (Region-CNN)

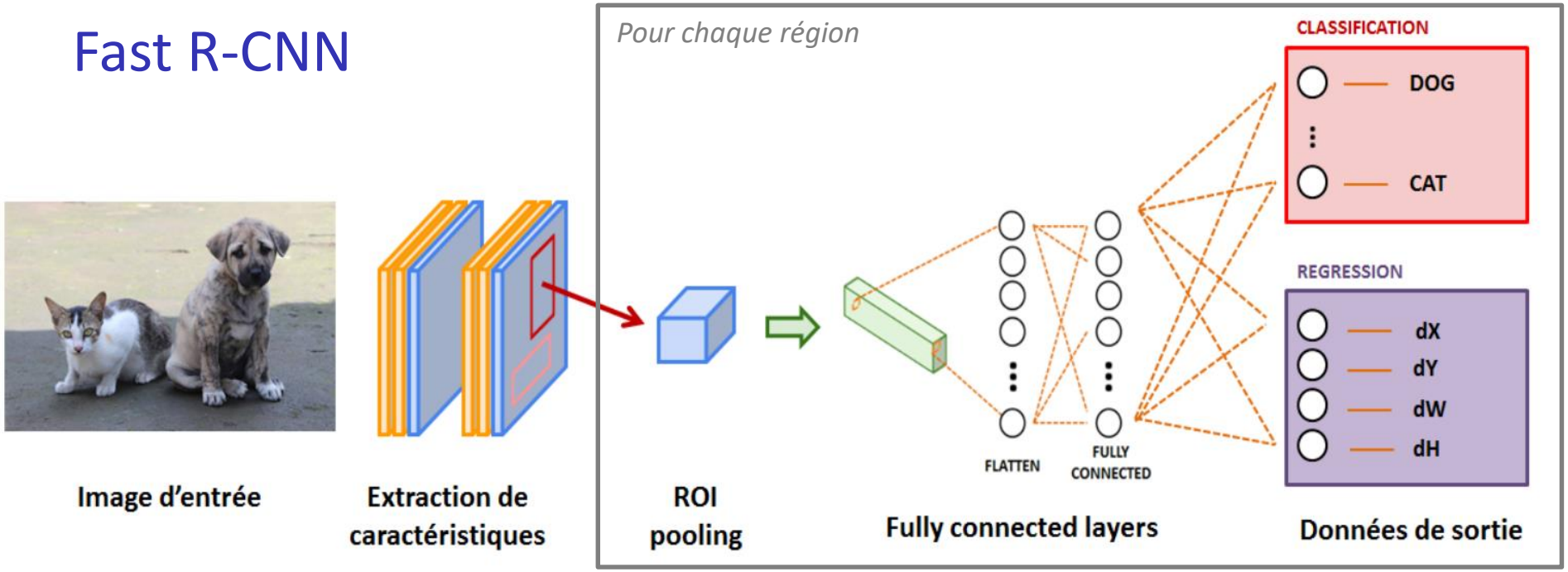
- Phase d'inférence



- Extraction des propositions de régions
- Inférences sur 2000 régions

Très lent ($\sim 50\ s$ par image)

Fast R-CNN

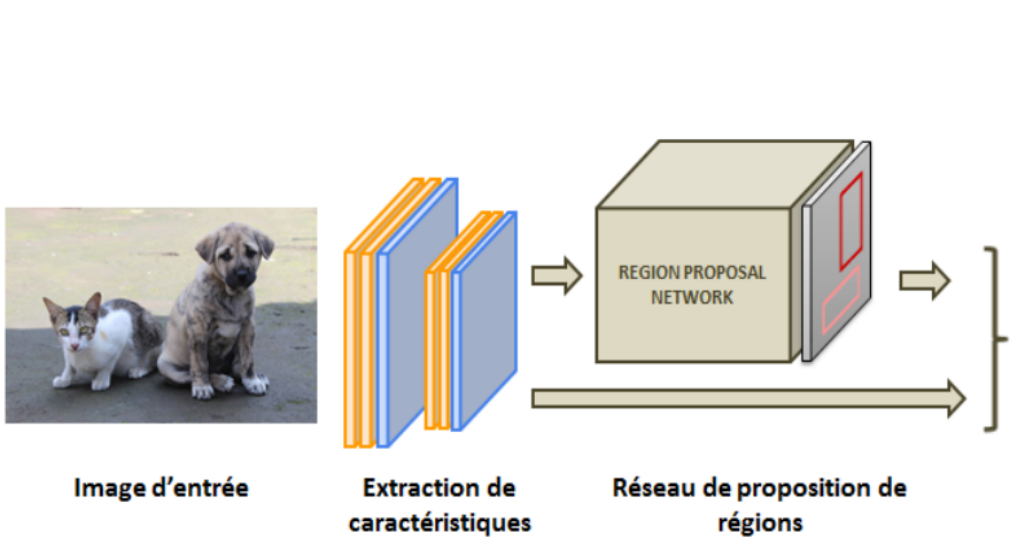


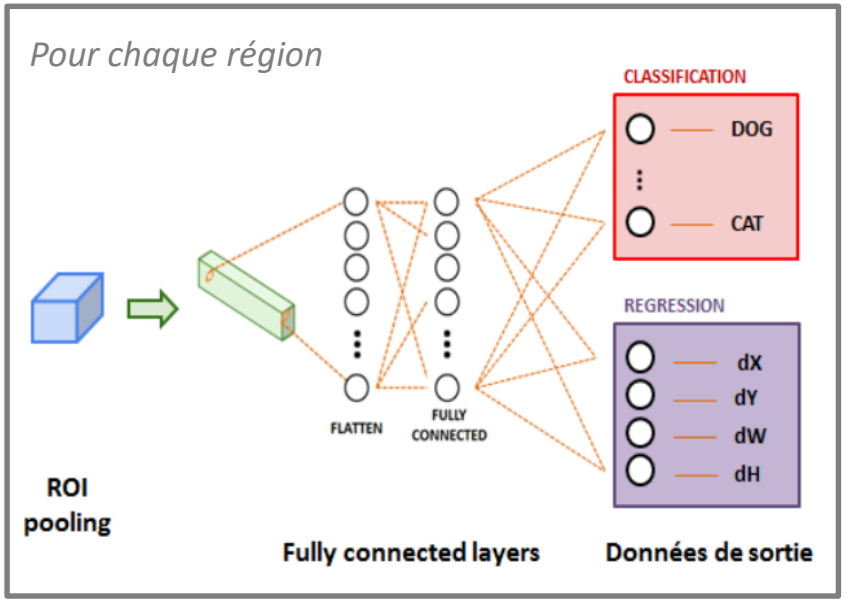
- 20x plus rapide que R-CNN en inférence !

Extraction de proposition de régions reste un point faible de la méthode

Girshick et al. "Fast R-CNN" ICCV, 2015

Faster R-CNN



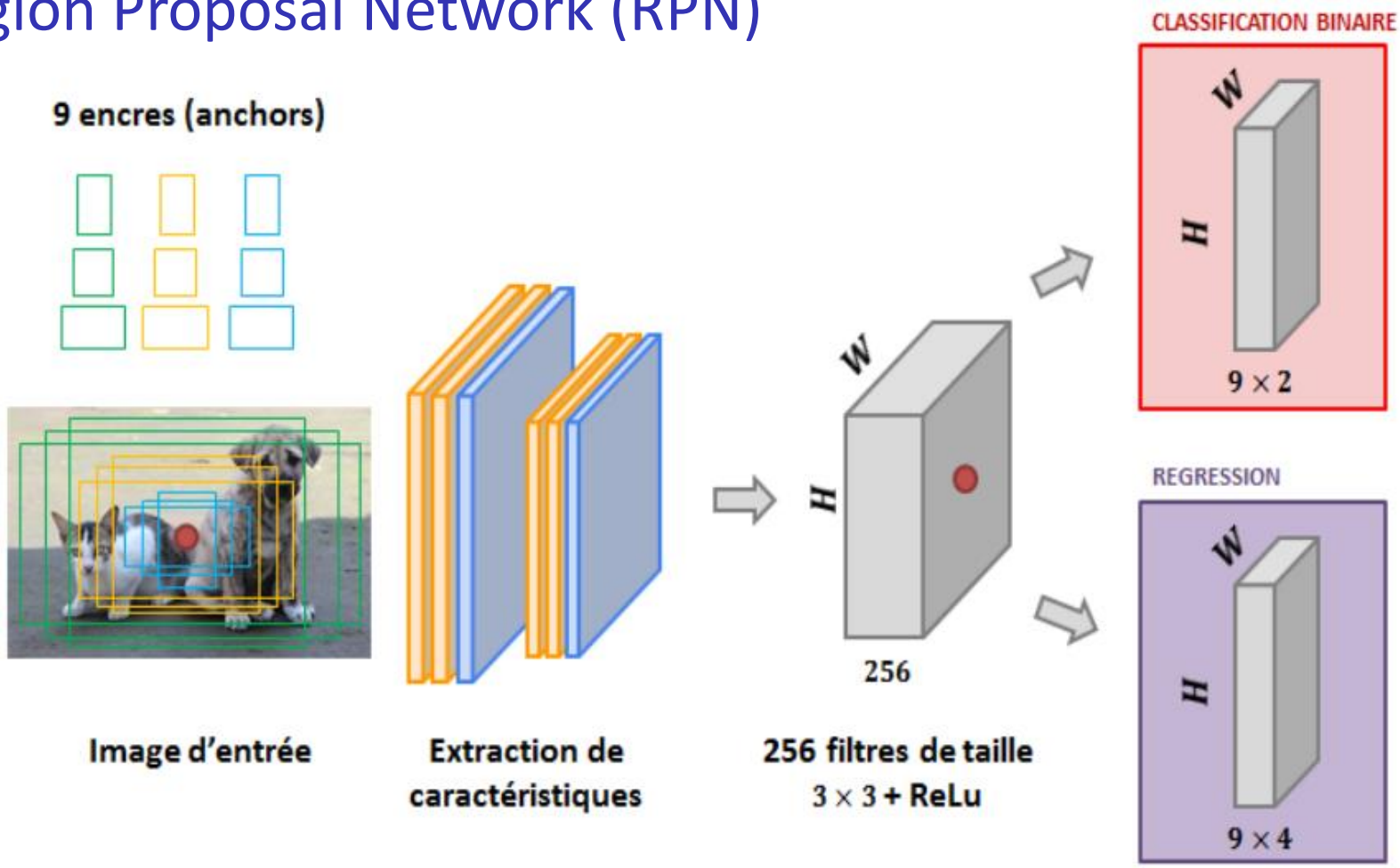


- Réseau de proposition de régions intégré

Réseau global entrainable de bout en bout !

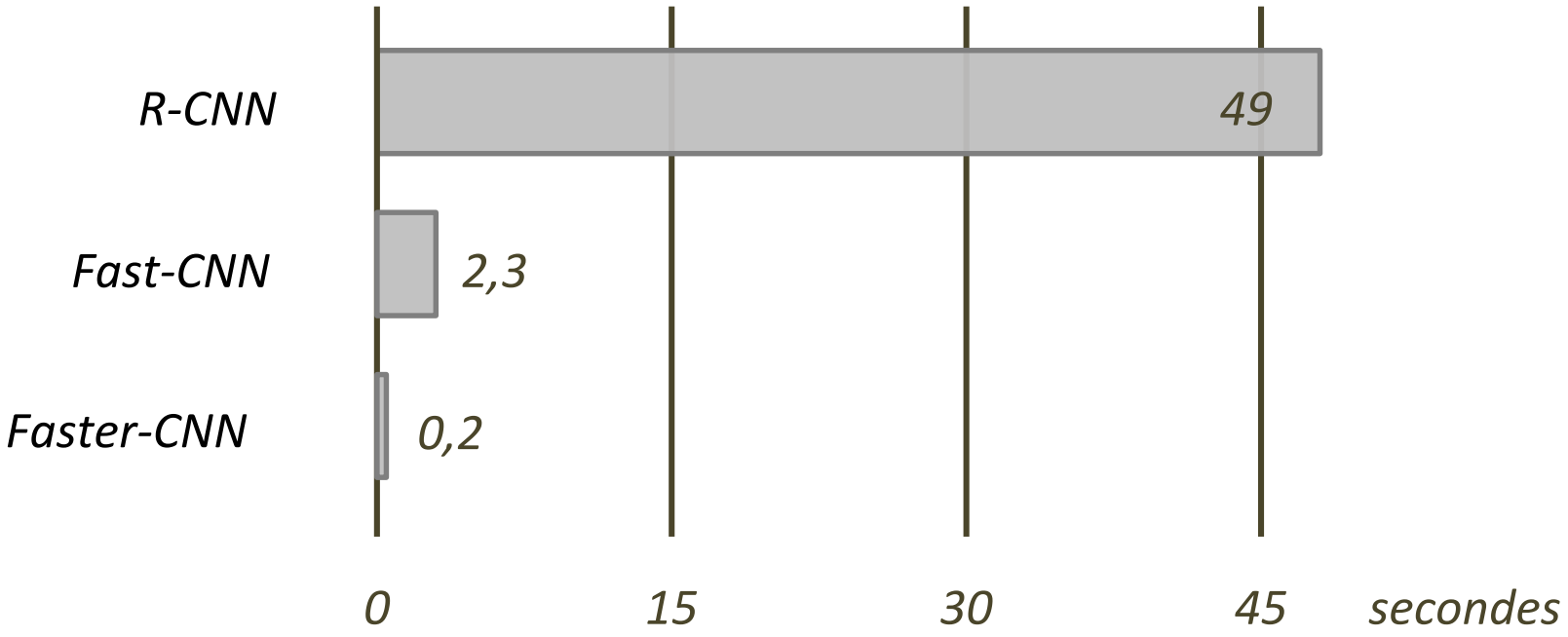
Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" NIPS, 2017

Region Proposal Network (RPN)



$W \times H \times 9$ régions détectées

Performance d'exécution (phase d'inférence)



Applications

Segmentation d'instances

Segmentation d'instances

Image d'entrée

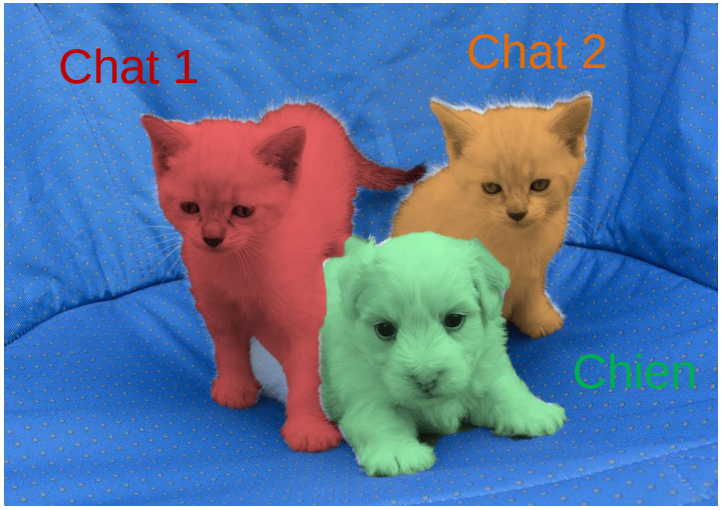
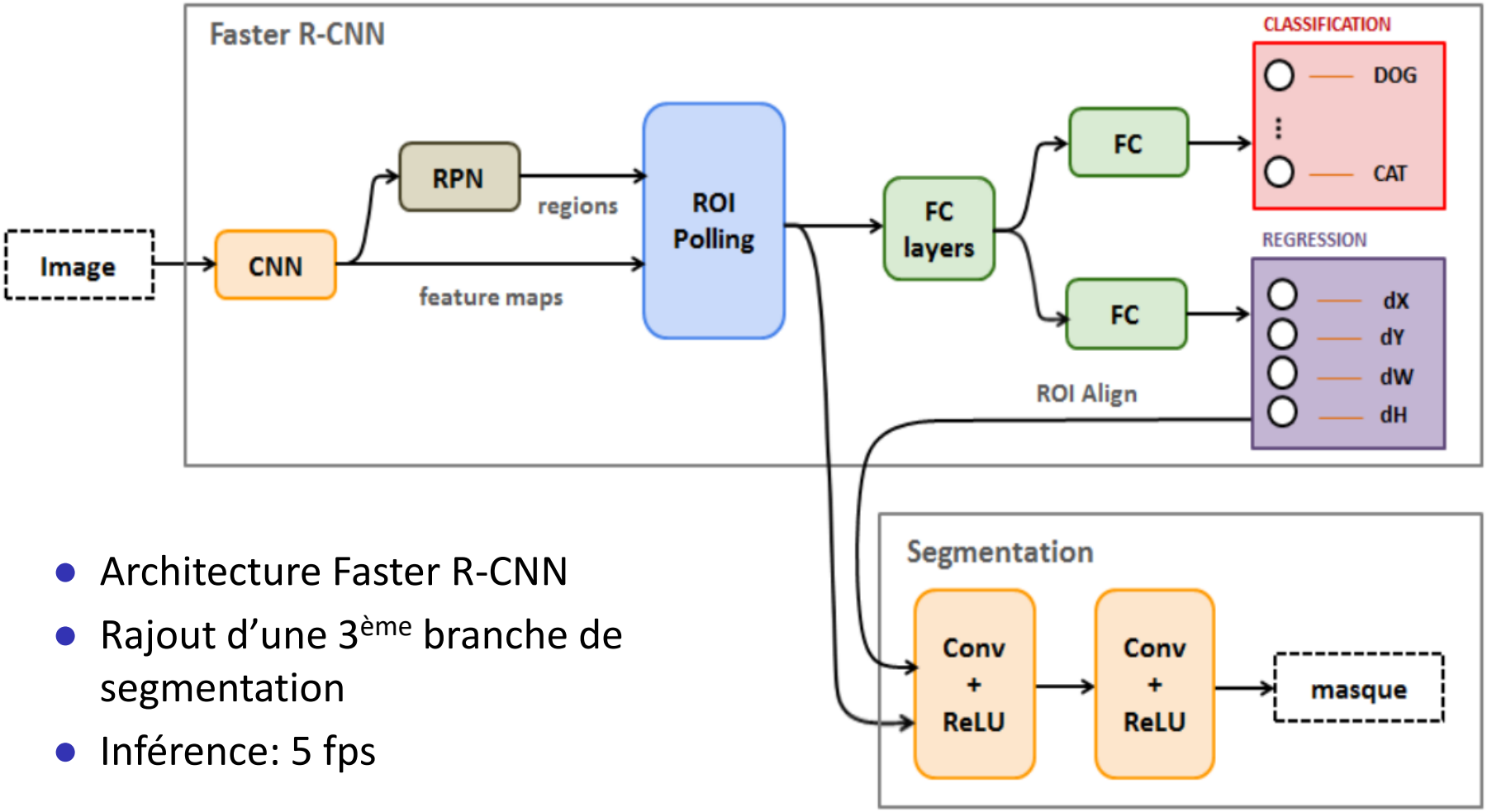


Image segmentée

Détecter et segmenter toutes les instances des objets/classes présents dans l'image

Mask R-CNN



- Architecture Faster R-CNN
- Rajout d'une 3ème branche de segmentation
- Inférence: 5 fps

He et al. "Mask R-CNN" ICCV, 2017

That’s all folks