

Réseaux de neurones

IFT 780

Segmentation et localisation

Par

Pierre-Marc Jodoin, Antoine Th  berge

1

---

---

---

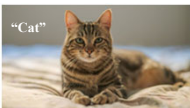


---

---

---

---

Classification



2

---

---

---




---

---

---

---

?



3

---

---

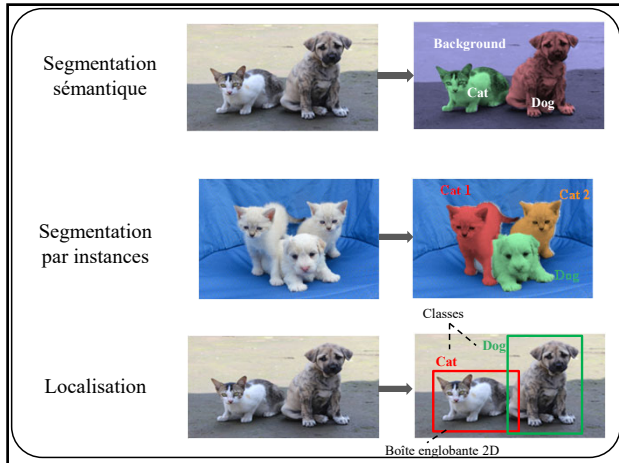
---

---

---

---

---



4

---

---

---

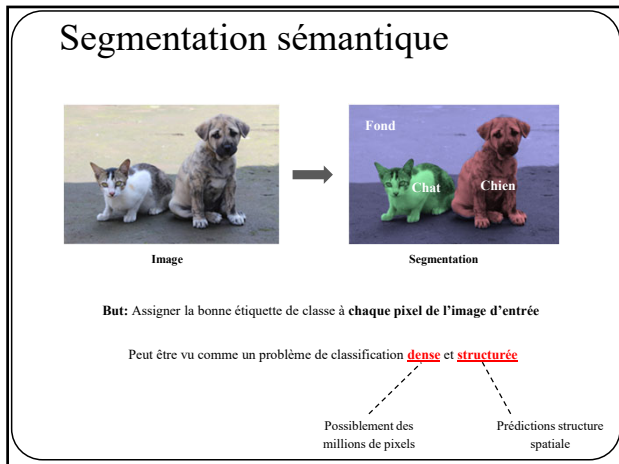
---

---

---

---

---



5

---

---

---

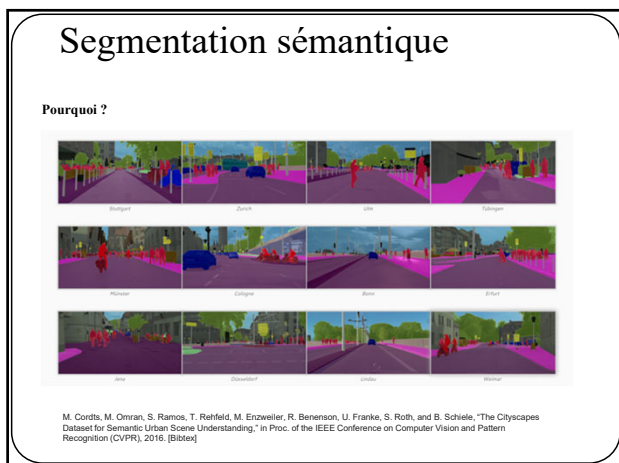
---

---

---

---

---



6

---

---

---

---

---



---

---

---

### Segmentation sémantique

Pourquoi ?



**Fig. 7:** Left: Original satellite image. Right: Semantic segmentation of roads, buildings and vegetation.

Ng, V., & Hofmann, D. (2018, July). Scalable feature extraction with aerial and satellite imagery. In Proceedings of the 17th Python in Science Conference (SCIPY 2018), Austin, TX, USA (pp. 9-15).

7

---

---

---

---

---

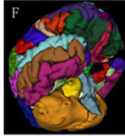
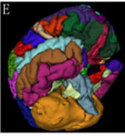
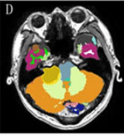
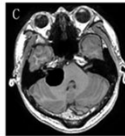

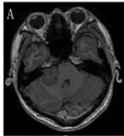
---

---

---

### Segmentation sémantique

Pourquoi ?



Hou, X., Yang, D., Li, D., Liu, M., Zhou, Y., & Shi, M. (2020). A new simple brain segmentation method for extracerebral intracranial tumors. *PLoS one*, 15(4), e0230754.

8

---

---

---

---

---

---

---

---


### Segmentation sémantique

Comment mesurer la performance de la segmentation ?

Pour la classification:

- "Top 1%"
- "Top 5%"
- ...

Pour la segmentation ?




Fond

Chat

Chien

Cible – Vérité terrain



Prédiction

9

---

---

---

---

---

---

---


---

### Segmentation sémantique

Comment mesurer la performance de la segmentation ?

Matrice de confusion:

		Vérité terrain	
		Positif	Négatif
Prédiction	Positif	True positive (TP)	False Positive (FP)
	Négatif	False negative (FN)	True negative (TN)



10

---

---

---

---

---

---

---

---


### Segmentation sémantique

Comment mesurer la performance de la segmentation ?

Justesse  
("Pixel accuracy")  $\frac{TP + TN}{TP + TN + FP + FN}$

Intersection over Union (IOU)/  
Jaccard Index  $\frac{TP}{TP + FP + FN}$

Dice  $\frac{2TP}{2TP + FP + FN}$



11

---

---

---

---

---

---

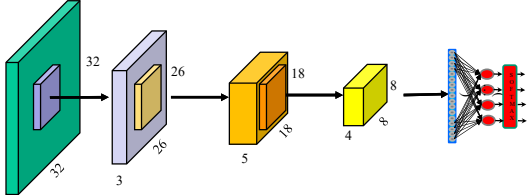
---

---

Image RGB : couche 1 : 3 filtres de taille 7x7  
couche 2 : 5 filtres de taille 9x9  
couche 3 : 4 filtres de taille 11x11  
convolution « valid »

Rappel  
classification  
d'images 32x32

Image: 32x32x3  
Stride : 1



3x(26x26) 2,028 neurones	5x(18x18) 1,620 neurones	4x(8x8) 256 neurones
3x7x7x3 441 paramètres	5x9x9x3 1,215 paramètres	4x11x11x5 2,420 paramètres

12

---

---

---

---

---

---

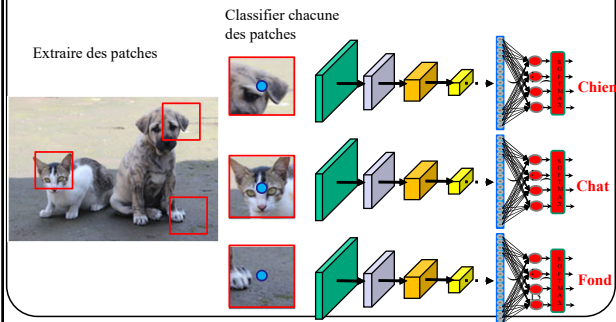
---

---

## Segmentation sémantique

Jusqu'à présent, on a vu comment classifier des images.

**Idée:** segmentation = classifier des sous-parties (*patches*) d'image

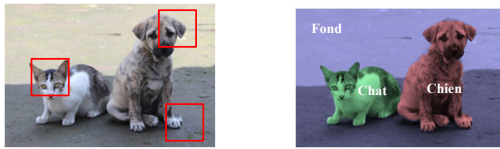


13

## Segmentation sémantique

Jusqu'à présent, on a vu comment classifier des images.

**Idée:** segmentation = classifier des sous-parties (*patches*) d'image



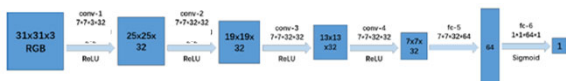
14

## Segmentation sémantique

Jusqu'à présent, on a vu comment classifier des images.

**Idée:** segmentation = classifier des sous-parties (*patches*) d'image

Exemple d'un réseau à convolution pour des patches RGB 31x31  
(Image tirée de l'article)



Wang Y, Luo Z., Jodoin P-M (2017) Interactive Deep Learning Method for Segmenting Moving Objects Pattern Recognition Letters, 96, p.66-75

15

## Plusieurs **inconvenients**

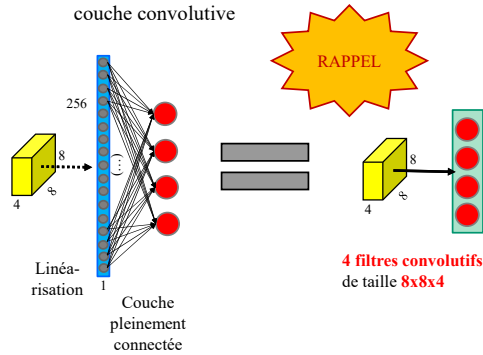
1. **Très long** tant en entraînement qu'en test
  1. Entraînement
    - Si 10,000 images 640x480 (300 000 pixels/image)
    - = 3 milliards de patches!
    - 1 epoch = 3 milliards de propagations avant et de rétro-propagations
2. Prédiction basée sur une **information locale (une patch)**

16

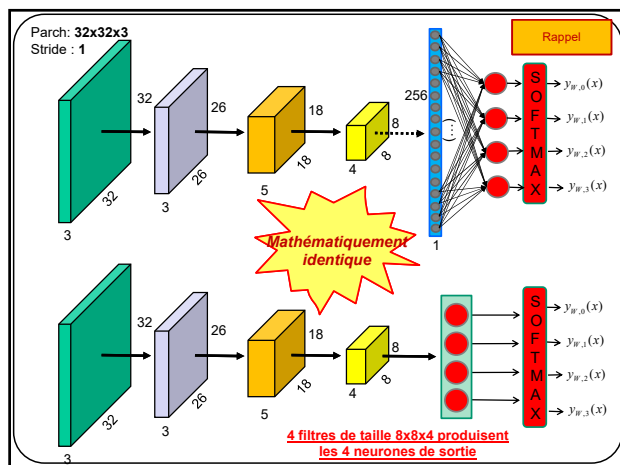
16

## Segmentation sémantique

**Amélioration 1:** remplacer la couche pleinement connectée par une couche convolutive



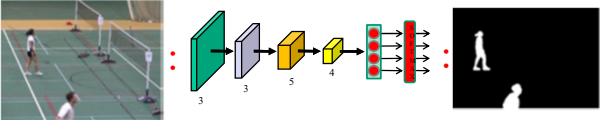
17



18

Image RGB : couche 1 : 3 filtres de taille 7x7  
couche 2 : 5 filtres de taille 9x9  
couche 3 : 4 filtres de taille 11x11  
convolution « valid »

Avec le réseau que voici, avec des conv « valid » et sans *pooling*, pour une image en **entrée de 320x240**, on aura en **sortie 289x209 pixels**, chacun ayant un vecteur de **4 prédictions**.



**Immense avantage** : fini les patches, on peut traiter une image avec 1 propagation avant et 1 rétropropagation

19

---

---

---

---

---

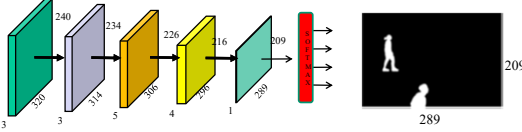
---

---

---

## Segmentation sémantique

Taille des cartes d'activation pour une image en entrée 320x240



**Immense avantage** : fini les patches, on peut traiter une image avec 1 propagation avant et 1 rétropropagation

20

---

---

---

---

---

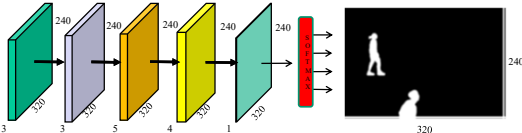
---

---

---

## Segmentation sémantique

Si on remplace les convolutions « valid » par des **convolutions « same »** (avec du *padding*) nous aurons en sortie une image de la même taille que l'image d'entrée



**Immense avantage** : fini les patches, on peut traiter une image avec 1 propagation avant et 1 rétropropagation

21

---

---

---

---

---

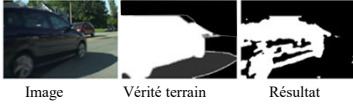
---

---

---

## Segmentation sémantique

Un réseau comme celui de la page précédente n'est jamais utilisé en pratique. Voici un exemple:



Pourquoi la prédiction est-elle bruitée ? Pourquoi autant de trous dans la prédiction ?

**Réponse : le "receptive field" est trop petit !**

Wang Y., Luo Z., Jodoin P.-M. (2017) Interactive Deep Learning Method for Segmenting Moving Objects Pattern Recognition Letters, 96, p.66-75

22

---

---

---

---

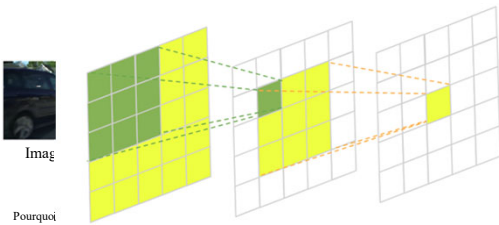
---

---

---

---

## Segmentation sémantique



**Réponse : le "receptive field" est trop petit !**

Wang Y., Luo Z., Jodoin P.-M. (2017) Interactive Deep Learning Method for Segmenting Moving Objects Pattern Recognition Letters, 96, p.66-75

23

---

---

---

---

---

---

---

---

## Note: taille du receptive field

$$r_0 = \sum_{l=1}^L \left( (k_l - 1) \prod_{i=1}^{l-1} s_i \right) + 1$$

$r$  = *receptive field*

$k$  = *kernel*

$s$  = *stride*

$l$  = *layers* du réseau

<https://distill.pub/2019/computing-receptive-fields>

24

---

---

---

---

---

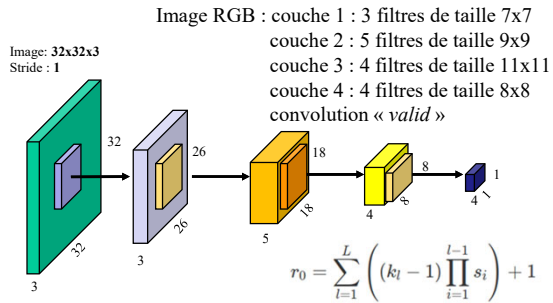
---

---

---



## Note: taille du receptive field

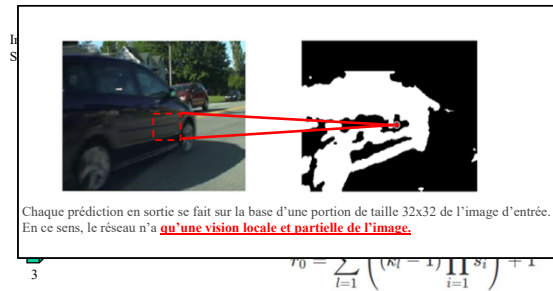


<https://distill.pub/2019/computing-receptive-fields>

$$6 + 8 + 10 + 7 + 1 = 32$$

25

## Note: taille du *receptive field*



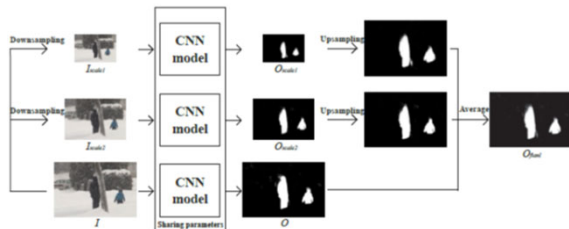
<https://distill.pub/2019/computing-receptive-fields>

$$6 + 8 + 10 + 7 + 1 = 32$$

26

## Segmentation sémantique

**Amélioration 2:** pour avoir plus de contexte dans la prédiction, entraîner un CNN avec des **images multirésolution**. En test, **combiner les prédictions** (ensemble de modèles)

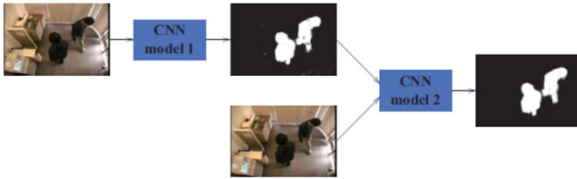


Wang Y, Luo Z., Jodoin P-M (2017) Interactive Deep Learning Method for Segmenting Moving Objects Pattern Recognition Letters, 96, p.66-75

27

## Segmentation sémantique

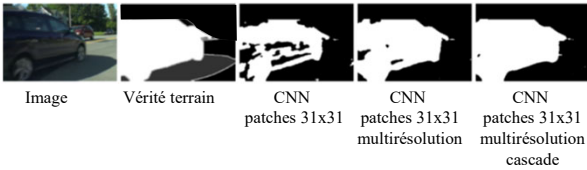
**Amélioration 3:** Pour raffiner les résultats, entraîner 2 modèles en cascade. Un premier qui segmente l'image d'entrée et le second qui segmente l'image d'entrée et la carte de segmentation du premier. Cela permet d'améliorer la cohésion spatiale.



Wang Y., Luo Z., Jodoin P-M (2017) Interactive Deep Learning Method for Segmenting Moving Objects Pattern Recognition Letters, 96, p.66-75

28

## Segmentation sémantique

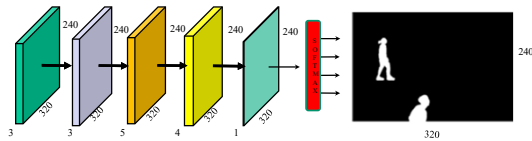


Wang Y., Luo Z., Jodoin P-M (2017) Interactive Deep Learning Method for Segmenting Moving Objects Pattern Recognition Letters, 96, p.66-75

29

## Segmentation sémantique

**Problème :** ce modèle a un champ récepteur (*receptive field*) relativement petit (ici 32x32). Au lieu, on aimerait que les pixels de sortie aient un champ récepteur de la taille de l'image d'entrée (ici 320x240)



**Solutions:**

- 1- ajouter beaucoup de couches
- 2- utiliser des convolutions dilatées (convolutions à trous)
- 3- mettre des couches de pooling après chaque bloc convolutionnel
- 4- faire un mélange de tout ça!

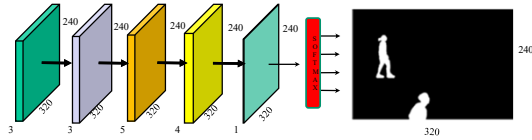
$$r_0 = \sum_{i=1}^L \left( (k_i - 1) \prod_{j=1}^{i-1} s_j \right) + 1$$

30

30

## Segmentation sémantique

**Problème** : ce modèle a un champ récepteur (*receptive field*) relativement petit (ici  $32 \times 32$ ).  
Au lieu, on aimerait que les pixels de sortie aient un champ récepteur de la taille de l'image d'entrée (ici  $320 \times 240$ )



**Solutions:**

- 1- ~~ajouter beaucoup de couches~~
- 2- utiliser des **convolutions dilatées** (convolutions *à trous*)
- 3- mettre des couches de pooling après chaque bloc convolutionnel
- 4- faire un mélange de tout ça!

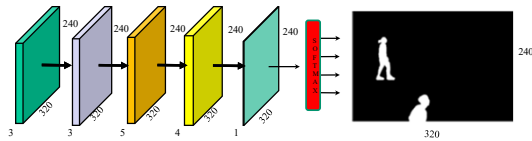
**Avec des filtres 3x3  
minimum de 120 couches!**

31

31

## Segmentation sémantique

**Problème** : ce modèle a un champ récepteur (*receptive field*) relativement petit (ici  $32 \times 32$ ).  
Au lieu, on aimerait que les pixels de sortie aient un champ récepteur de la taille de l'image d'entrée (ici  $320 \times 240$ )



**Solutions:**

- 1- ~~ajouter beaucoup de couches~~
- 2- utiliser des **convolutions dilatées** (convolutions *à trous*)
- 3- mettre des couches de pooling après chaque bloc convolutionnel
- 4- faire un mélange de tout ça!



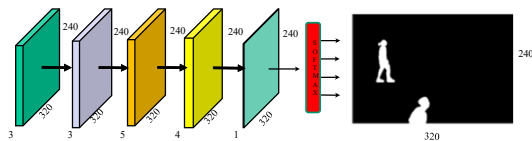
**Explosion de la mémoire  
Et des temps de calculs  
Et problème de disparition  
de gradients**

32

32

## Segmentation sémantique

**Problème** : ce modèle a un champ récepteur (*receptive field*) relativement petit (ici  $32 \times 32$ ).  
Au lieu, on aimerait que les pixels de sortie aient un champ récepteur de la taille de l'image d'entrée (ici  $320 \times 240$ )



**Solutions:**

- 1- ~~ajouter beaucoup de couches~~
- 2- utiliser des **convolutions dilatées** (convolutions *à trous*)
- 3- mettre des couches de pooling après chaque bloc convolutionnel
- 4- faire un mélange de tout ça!

33

33

### Segmentation sémantique

Rappel

Noyau 3x3

Convolution usuelle

Noyau 3x3

À trous  
(taux = 2)

Noyau 3x3

À trous  
(taux = 3)

Champ récepteur=3x3    Champ récepteur=5x5    Champ récepteur=7x7

34

---

---

---

---

---

---

---

---

34

### Segmentation sémantique

→ Conv 3x3 → Conv 3x3 → Conv 3x3 → Conv 3x3 →

Champ récepteur (taux = 1)	3x3	5x5	7x7	9x9
Champ récepteur (taux = 2)	5x5	9x9	13x13	17x17
Champ récepteur (taux = 3)	7x7	13x13	19x19	21x21

$$r_0 = \sum_{i=1}^L \left( (k_i - 1) \prod_{j=1}^{i-1} s_j \right) + 1 \quad k = \text{taux} * (k-1) + 1$$

35

---

---

---

---

---

---

---

---

35

### Segmentation sémantique

→ Conv 5x5 → Conv 5x5 → Conv 5x5 → Conv 5x5 →

Champ récepteur (taux = 1)	5x5	9x9	13x13	17x17
Champ récepteur (taux = 2)	9x9	17x17	25x25	33x33
Champ récepteur (taux = 3)	15x15	29x29	43x43	57x57

$$r_0 = \sum_{i=1}^L \left( (k_i - 1) \prod_{j=1}^{i-1} s_j \right) + 1 \quad k = \text{taux} * (k-1) + 1$$

36

---

---

---

---

---

---

---

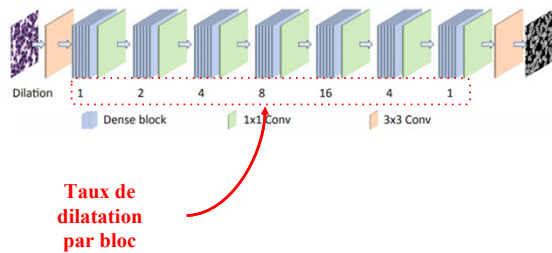
---

36

12

## FullNet

Le « **FullNet** » [Qu et al. 2019] implémente ce type de réseau mais avec des blocs convolutifs **denses** comme ceux du **denseNet**.



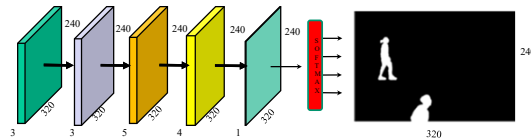
H. Qu, Z. Yan, G.M. Riedlinger, S. De and D.N. Metaxas  
"Improving Nuclei/Gland Instance Segmentation in Histopathology Images  
by Full Resolution Neural Network and Spatial Constrained Loss", in proc of MICCAI 2019

37

37

## Segmentation sémantique

**Problème** : ce modèle a un champ récepteur (*receptive field*) relativement petit (ici  $32 \times 32$ ). Au lieu, on aimerait que les pixels de sortie aient un champ récepteur de la taille de l'image d'entrée (ici  $320 \times 240$ ).



### Solutions:

- 1- ajouter beaucoup de couches
- 2- utiliser des convolutions dilatées (convolutions à trous)
- 3- mettre des couches de **pooling après chaque bloc convolutionnel**
- 4- faire un mélange de tout ça!

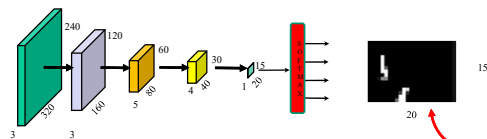
Image: Long et al. "Fully convolutional networks for semantic segmentation." ICCV, 2015.

38

38

## Segmentation sémantique

**Problème** : ce modèle a un champ récepteur (*receptive field*) relativement petit (ici  $32 \times 32$ ). Au lieu, on aimerait que les pixels de sortie aient un champ récepteur de la taille de l'image d'entrée (ici  $320 \times 240$ ).



### Solutions:

- 1- ajouter beaucoup de couches
- 2- utiliser des convolutions dilatées (convolutions à trous)
- 3- mettre des couches de **pooling après chaque bloc convolutionnel**
- 4- faire un mélange de tout ça!

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

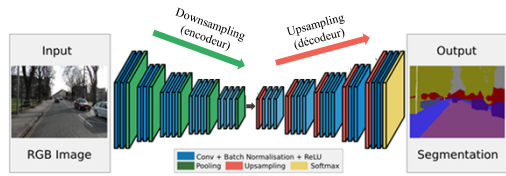
39

39

## Segmentation sémantique

**Solution** : augmenter la resolution en sortie à l'aide d'un **décodateur**.

Réseau encodeur-décodateur (ici "SegNet")



**Problème**: la résolution spatiale est perdue avec les couches de sous-échantillonnage (**Conv + Pooling**)

**Solution**: Augmenter la resolution à l'aide d'un décodeur et de couches de sur-échantillonnage (**??? + Conv**)

Adapté de: Badrinarayanan et al. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." PAMI, 2017. 40

40

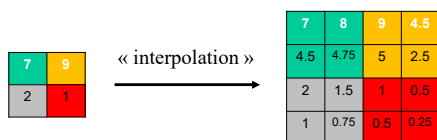
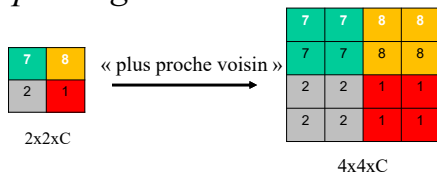
Pour **augmenter la taille** des cartes d'activation il faut une opération de "**upsampling**"

Deux types d'approches

- Méthodes sans paramètres => unpooling
- Méthode avec paramètres => convolution transposée

41

## Unpooling



42

# Convolution transposée

L'idée ici est moins intuitive que pour du unpooling.  
**Commençons par un exemple 1D...**

43

---

---

---

---

---

---

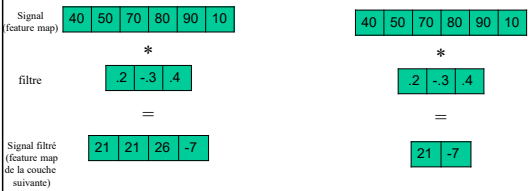
---

# Convolution de base

(exemple 1d)

Convolution "valid" stride =1

Convolution "valid" stride =3



44

---

---

---

---

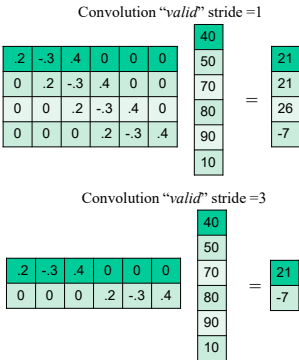
---

---

---

# Opération matrice-vecteur

(exemple 1d)



45

---

---

---

---

---

---

---

## Opération matrice-vecteur

(exemple 1d)

Convolution "valid" stride =1

$$\begin{array}{|c|c|c|c|c|c|} \hline 2 & -3 & 4 & 0 & 0 & 0 \\ \hline 0 & 2 & -3 & 4 & 0 & 0 \\ \hline 1 & 0 & 0 & 2 & -3 & 4 \\ \hline \end{array} \begin{array}{|c|} \hline 40 \\ \hline 50 \\ \hline 70 \\ \hline 80 \\ \hline 90 \\ \hline 10 \\ \hline \end{array} = \begin{array}{|c|} \hline 21 \\ \hline 26 \\ \hline -7 \\ \hline \end{array}$$

$4 \times 6 \quad 6 \times 1 = 4 \times 1$

Convolution "valid" stride =3

$$\begin{array}{|c|c|c|c|c|c|} \hline 2 & -3 & 4 & 0 & 0 & 0 \\ \hline 0 & 2 & -3 & 4 & 0 & 0 \\ \hline 1 & 0 & 0 & 2 & -3 & 4 \\ \hline \end{array} \begin{array}{|c|} \hline 40 \\ \hline 50 \\ \hline 70 \\ \hline 80 \\ \hline 90 \\ \hline 10 \\ \hline \end{array} = \begin{array}{|c|} \hline 21 \\ \hline 26 \\ \hline -7 \\ \hline \end{array}$$

$2 \times 6 \quad 6 \times 1 = 2 \times 1$

46

46

---

---

---

---

---

---

---

---

## Convolution transposée

Le but est d'avoir un filtre dont la taille des opérations **est inversée**

Convolution "valid"  
stride =1

$$\begin{array}{|c|c|c|c|c|} \hline 2 & 0 & 0 & 0 & 0 \\ \hline -3 & 2 & 0 & 0 & 0 \\ \hline 4 & -3 & 2 & 0 & 0 \\ \hline 0 & 4 & -3 & 2 & 0 \\ \hline 0 & 0 & 4 & -3 & 2 \\ \hline 0 & 0 & 0 & 4 & -3 \\ \hline \end{array} \begin{array}{|c|} \hline 40 \\ \hline 50 \\ \hline 70 \\ \hline 80 \\ \hline \end{array} = \begin{array}{|c|} \hline 8 \\ \hline -2 \\ \hline 15 \\ \hline 15 \\ \hline 4 \\ \hline 32 \\ \hline \end{array}$$

Convolution "valid"  
stride =3

$$\begin{array}{|c|c|} \hline 2 & 0 \\ \hline -3 & 0 \\ \hline 4 & 0 \\ \hline 0 & 2 \\ \hline 0 & -3 \\ \hline 0 & 4 \\ \hline \end{array} \begin{array}{|c|} \hline 40 \\ \hline 50 \\ \hline \end{array} = \begin{array}{|c|} \hline 8 \\ \hline -12 \\ \hline -16 \\ \hline 10 \\ \hline -15 \\ \hline 20 \\ \hline \end{array}$$

47

47

---

---

---

---

---

---

---

---

## Convolution transposée

Le but est d'avoir un filtre dont la taille des opérations **est inversée**

Convolution "valid"  
stride =1

$$\begin{array}{|c|c|c|c|c|} \hline 2 & 0 & 0 & 0 & 0 \\ \hline -3 & 2 & 0 & 0 & 0 \\ \hline 4 & -3 & 2 & 0 & 0 \\ \hline 0 & 4 & -3 & 2 & 0 \\ \hline 0 & 0 & 4 & -3 & 2 \\ \hline 0 & 0 & 0 & 4 & -3 \\ \hline \end{array} \begin{array}{|c|} \hline 40 \\ \hline 50 \\ \hline 70 \\ \hline 80 \\ \hline \end{array} = \begin{array}{|c|} \hline 8 \\ \hline -2 \\ \hline 15 \\ \hline 15 \\ \hline 4 \\ \hline 32 \\ \hline \end{array}$$

$6 \times 4 \quad 4 \times 1 = 6 \times 1$

Convolution "valid"  
stride =3

$$\begin{array}{|c|c|} \hline 2 & 0 \\ \hline -3 & 0 \\ \hline 4 & 0 \\ \hline 0 & 2 \\ \hline 0 & -3 \\ \hline 0 & 4 \\ \hline \end{array} \begin{array}{|c|} \hline 21 \\ \hline 26 \\ \hline -7 \\ \hline \end{array} = \begin{array}{|c|} \hline 8 \\ \hline -12 \\ \hline -16 \\ \hline 10 \\ \hline -15 \\ \hline 20 \\ \hline \end{array}$$

$6 \times 2 \quad 2 \times 1 = 6 \times 1$

48

48

---

---

---

---

---

---

---

---



## Convolution transposée

Le but est d'avoir un filtre dont la taille des opérations **est inversée**

Convolution "valid"  
stride = 1

.2	0	0	0
-.3	.2	0	0
.4	-.3	.2	0
0	.4	-.3	.2
0	0	.4	-.3
0	0	0	.4

Convolution "valid"  
stride = 3

.2	0
-.3	0
.4	0
0	.2
0	-.3
0	.4

Matrices  
transposées

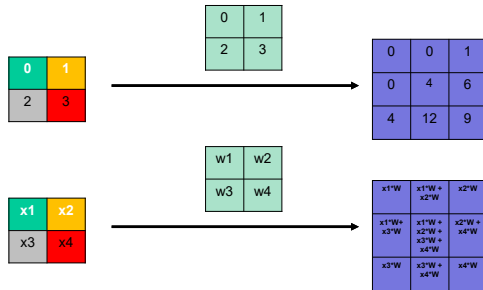
49

49

## Convolution transposée

Exemple avec  
2x2 stride 1

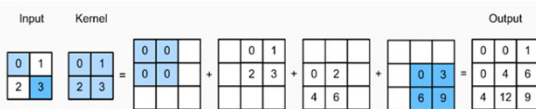
Stride dans l'espace de "sortie", pas l'espace "d'entrée"



50

## Convolution transposée

Exemple chiffré avec 2x2 stride 1



Autres noms:

- Déconvolution
- Upconvolution
- Fractionally-strided convolution

Ressemble  
beaucoup à la  
rétropropagation  
d'une couche  
convolutive !

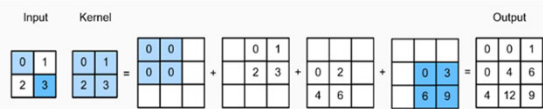
<https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba>

51

51

## Convolution transposée

Exemple chiffré avec 2x2 stride 1



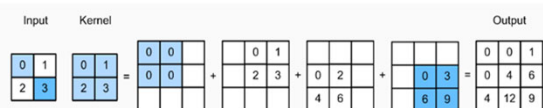
**Problème ?**

<https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba>

52

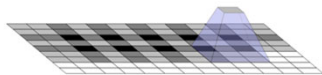
## Convolution transposée

Exemple chiffré avec 2x2 stride 1



**Problème ?**

Crée des artefacts de "damier" au chevauchement (*checkerboard*)



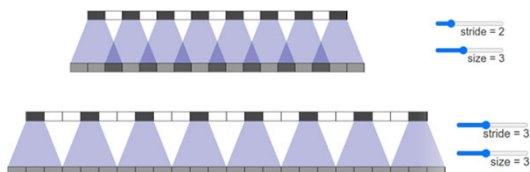
<https://distill.pub/2016/deconv-checkerboard/>

53

## Convolution transposée

**Solution ? (partielle)**

Augmenter le *stride* pour éviter le chevauchement



<https://distill.pub/2016/deconv-checkerboard/>

54

## Convolution transposée

### Solution ? (partielle)

Augmenter le *stride*  
pour éviter le chevauchement.



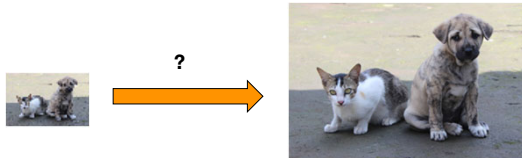
<https://distill.pub/2016/deconv-checkerboard/>

55

55

## Note: super-résolution

Comment entraîner un réseau à convolutions à faire de la super-résolution ?

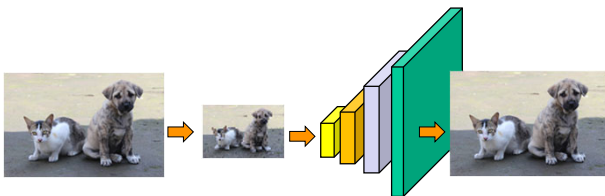


58

58

## Note: super-résolution

S'entraîner avec des images à haute-résolution qui ont été réduites !

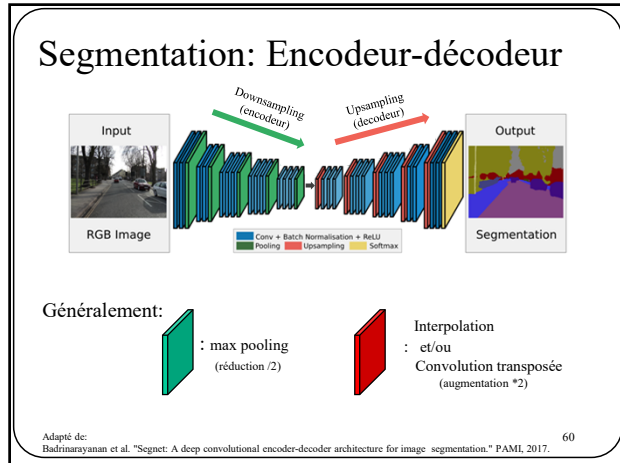


Sub-pixel ou interpolation + convolution

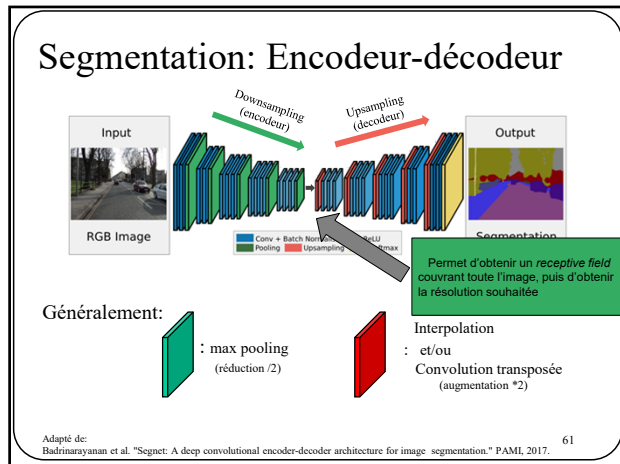
Shi, W., Caballero, J., Huszar, F., Totz, J., Aitken, A. P., Bishop, R., ... & Wang, Z. (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1874-1883).

59

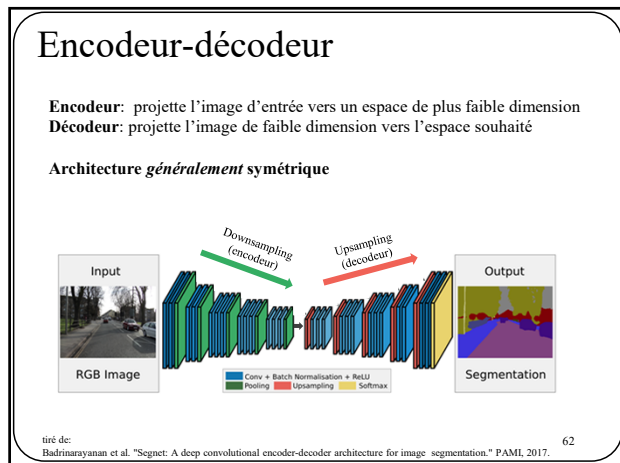
59



60



61



62

## Encodeur-décodeur

**Encodeur:** projette l'image d'entrée vers un espace de plus faible dimension  
**Décodeur:** projette l'image de faible dimension vers l'espace souhaité

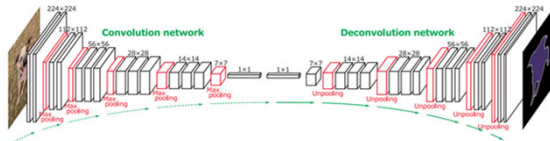
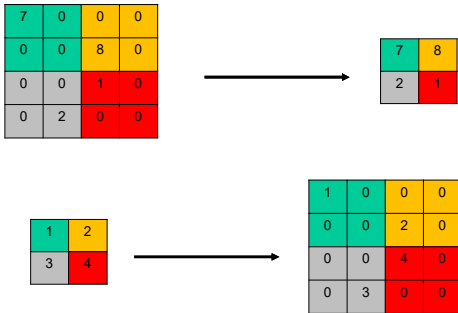


Figure 2. Overall architecture of the proposed network. On top of the convolution network based on VGG 16-layer net, we put a multi-layer deconvolution network to generate the accurate segmentation map of an input proposal. Given a feature representation obtained from the convolution network, dense pixel-wise class prediction map is constructed through multiple series of unpooling, deconvolution and rectification operations.

tiré de: Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 1520-1528). 63

63

## Max pooling - Max unpooling



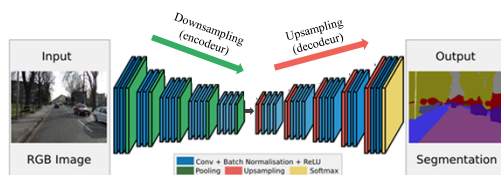
Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision* (pp. 1520-1528).

64

## Encodeur-décodeur

Un inconvénient des structures encodeur-décodeur

Perte d'information

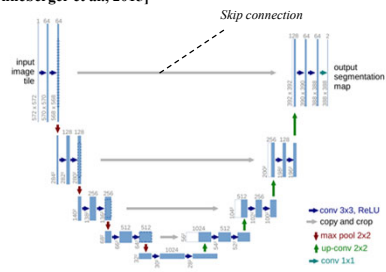


Adapté de: Badrinarayanan et al. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." PAMI, 2017. 65

65

## Solution : les *skip connections*

U-Net [Ronneberger et al., 2015]



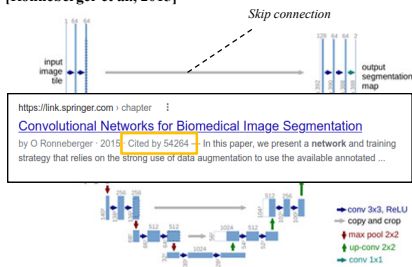
CNN le plus populaire en imagerie médicale

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

66

## Solution : les *skip connections*

U-Net [Ronneberger et al., 2015]

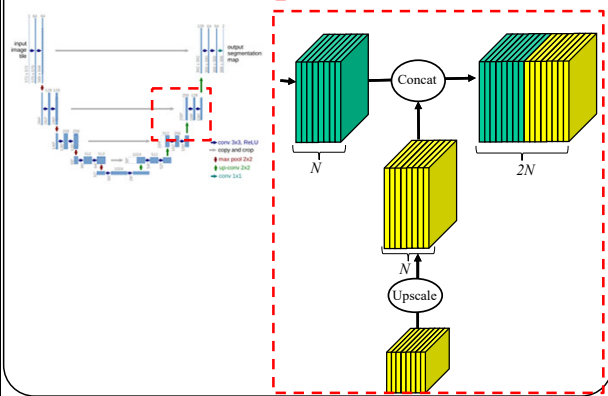


CNN le plus populaire en imagerie médicale

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

67

## Solution : les *skip connections*



68

## 3D-UNet/V-Net

Identiques au Unet mais avec des **convolutions 3D**

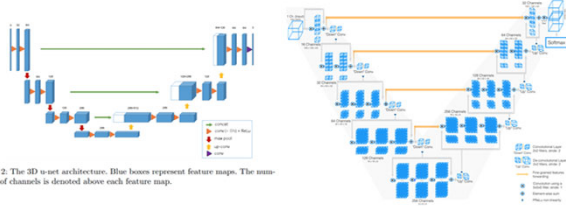


Fig. 2 The 3D u-net architecture. Blue boxes represent feature maps. The number of channels is denoted above each feature map.

Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016, October). 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention* (pp. 424-432). Springer, Cham.

Milletari, F., Navab, N., & Ahmadi, S. A. (2016, October). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)* (pp. 565-571). IEEE.

69

## 3D-UNet/V-Net

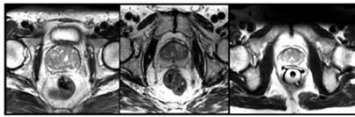


Fig.1. Slices from MRI volumes depicting prostate. This data is part of the PROMISE2012 challenge dataset [7].



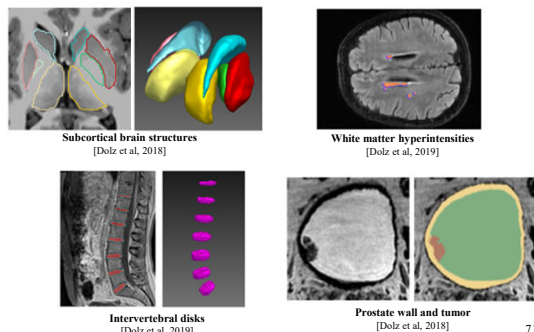
Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016, October). 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention* (pp. 424-432). Springer, Cham.

Milletari, F., Navab, N., & Ahmadi, S. A. (2016, October). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)* (pp. 565-571). IEEE.

70

## Imagerie médicale

Exemples d'images 3D en imagerie médicale



Subcortical brain structures  
[Dolz et al, 2018]

White matter hyperintensities  
[Dolz et al, 2019]

Intervertebral disks  
[Dolz et al, 2019]

Prostate wall and tumor  
[Dolz et al, 2018]

71

71





E-Net  
(E pour *Efficient*)

Table 2: Performance comparison.

Model	NVIDIA TX1						NVIDIA Titan X					
	480×320		640×360		1280×720		640×360		1280×720		1920×1080	
	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps	ms	fps
SegNet	757	1.3	1251	0.8	-	-	69	14.6	289	3.5	637	1.6
E-Net	47	21.1	69	14.6	262	3.8	7	135.4	21	46.8	46	21.6

Table 3: Hardware requirements. FLOPs are estimated for an input of 3 × 640 × 360.

	GFLOPs	Parameters	Model size (fp16)
SegNet	286.03	29.46M	56.2 MB
E-Net	3.83	0.37M	0.7 MB

Très efficace!!! 300 fois moins de calculs pour des résultats similaires à SegNet

Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation, *arXiv: 1606.02147*, 2016.

75

DeepLab V1,V2,V3, PSPNet, MSCADC, etc.

Plusieurs méthodes utilisent à la fois des **convolutions dilatées** et du « **upsampling** ».

Configuration typique:

```
graph LR; Input[Input Image] --> Backend[Backend Vgg ou ResNet]; Backend --> Dilated[Quelques conv dilatéés]; Dilated --> Upsampling[Upsampling]; Upsampling --> Output[Output Mask]
```

H.Zhao, J.Shi, X. Qi, X. Wang, J. Jia, Pyramid Scene Parsing Network, CVPR 2017

L. Chen, G. Papandreou, I.Kokkinos, K.Murphy, A.L. Yuille Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs, ICLR 2015

F. Yu, V. Koltun, Multi-Scale Context Aggregation by Dilated Convolution, ICLR 2016

L. Chen, G. Papandreou, I.Kokkinos, K.Murphy, A.L. Yuille DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, PAMI 2016

76

DeepLab V1,V2,V3, PSPNet, MSCADC, etc.

Une méthode très populaire : **DeepLab**

```
graph LR; Input[Input Image] --> Backend[Backend Vgg ou ResNet]; Backend --> Dilated[Quelques conv dilatéés]; Dilated --> Upsampling[Upsampling]; Upsampling --> Output[Output Mask]
```

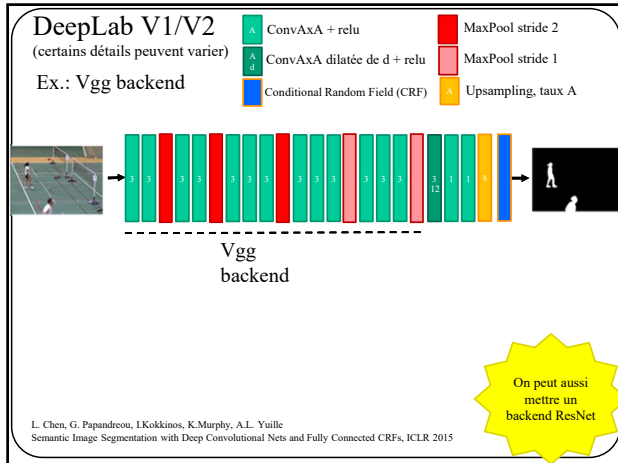
H.Zhao, J.Shi, X. Qi, X. Wang, J. Jia, Pyramid Scene Parsing Network, CVPR 2017

L. Chen, G. Papandreou, I.Kokkinos, K.Murphy, A.L. Yuille Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs, ICLR 2015

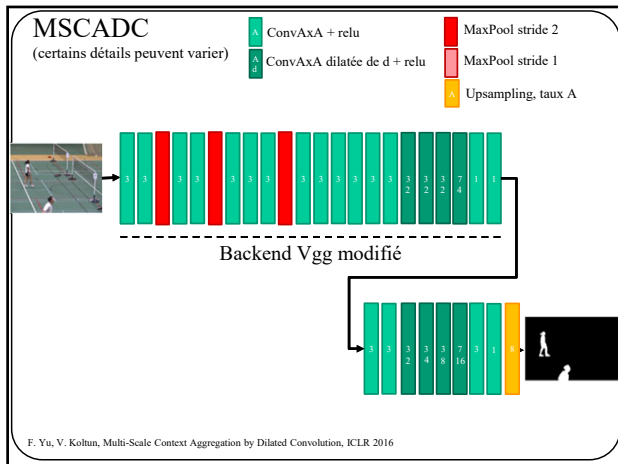
F. Yu, V. Koltun, Multi-Scale Context Aggregation by Dilated Convolution, ICLR 2016

L. Chen, G. Papandreou, I.Kokkinos, K.Murphy, A.L. Yuille DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, PAMI 2016

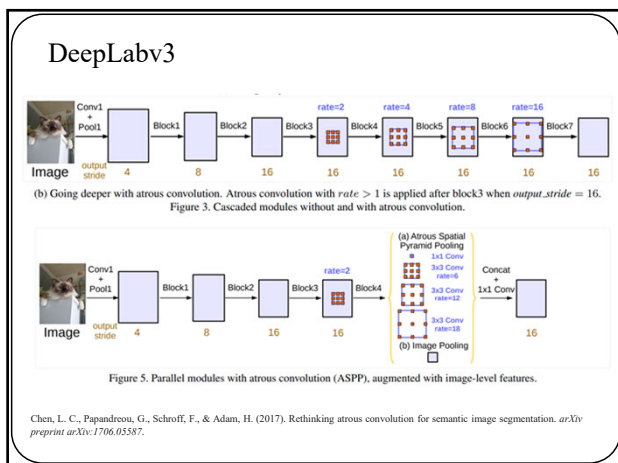
77



78



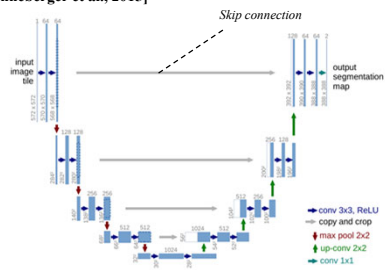
79



80

## En pratique:

U-Net [Ronneberger et al., 2015]



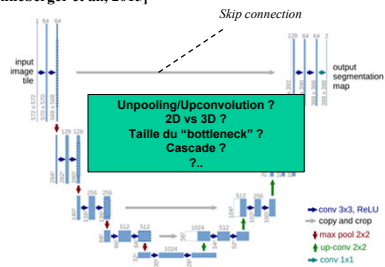
CNN le plus populaire en imagerie médicale

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

81

## En pratique:

U-Net [Ronneberger et al., 2015]



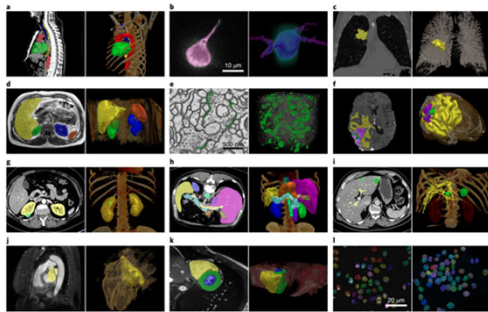
CNN le plus populaire en imagerie médicale

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

82

## En pratique : nnU-Net

L'imagerie médicale est un domaine vaste et varié

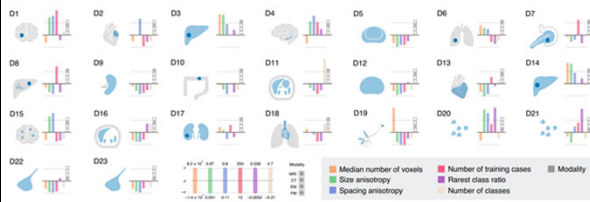


Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature methods, 18(2), 203-211.

83

## En pratique : nnU-Net

L'imagerie médicale est un domaine vaste et varié

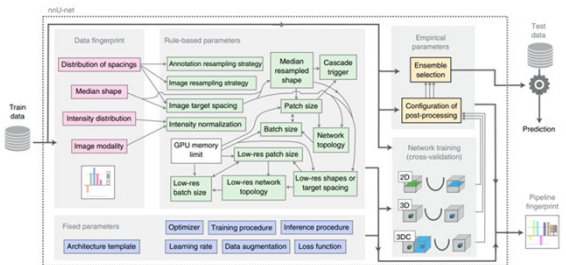


Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2), 203-211.

84

## En pratique : nnU-Net (*no-new-net*)

L'imagerie médicale est un domaine vaste et varié

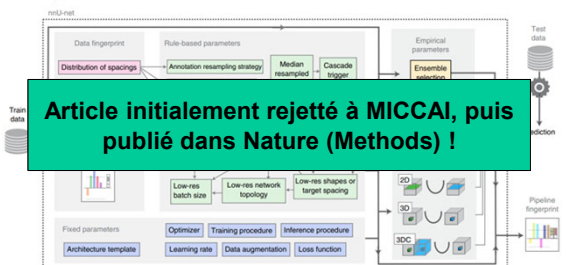


Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2), 203-211.

85

## En pratique : nnU-Net (*no-new-net*)

L'imagerie médicale est un domaine vaste et varié



Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2), 203-211.

86

## En pratique : nnU-Net (*no-new-net*)

**nnU-Net est de loin le meilleur!**

### nnU-Net Application: Out-of-the-box Quantitative Results

Evaluation performed on 23 datasets from biomedical segmentation competitions



Beyond the Patterns 29 - Fabian Isensee - nnU-Net: self-configuring deep learning image segmentation  
<https://www.youtube.com/watch?v=C6tpnJRpt90>

Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2), 203-211.

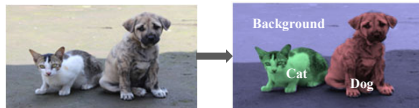
87

## DÉTECTION D'OBJETS

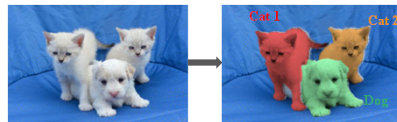
88

88

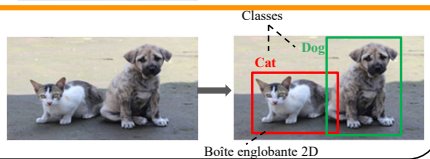
Segmentation  
sémantique



Segmentation  
par instances

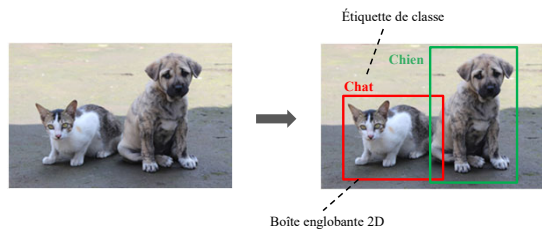


Localisation  
/ détection



89

## Détection d'objets



**BUT:** localiser et reconnaître des objets (ou personnes, animaux).

90

90

---

---

---

---

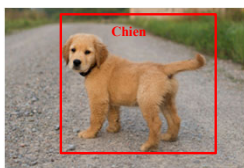
---

---

---

---

## Détection d'un seul objet



91

91

---

---

---

---

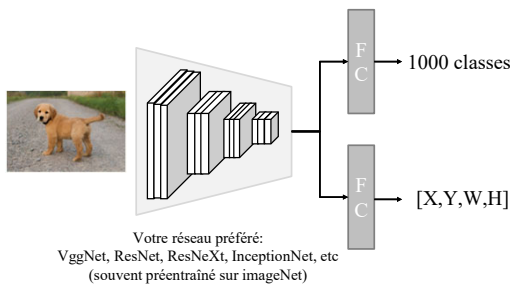
---

---

---

---

## Détection d'un seul objet



92

92

---

---

---

---

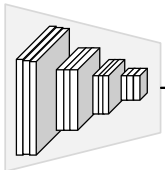

---

---

---

---

Détection d'un seul objet



« Backend »

FC

1000 classes

FC

[X,Y,W,H]

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, InceptionNet, etc  
(souvent préentraîné sur imageNet)

93

---

---

---

---

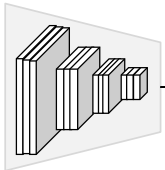

---

---

---

---

Détection d'un seul objet



FC

1000 classes

FC

[X,Y,W,H]

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, InceptionNet, etc  
(souvent préentraîné sur imageNet)

Deux têtes de sortie :

1- Classification

2- Régression

94

---

---

---

---

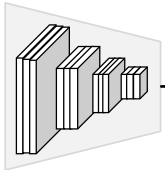

---

---

---

---

Détection d'un seul objet



« Backend »

FC

Classe cible : 'Chien'

↓

Loss classification  
(Entropie croisée ou autre)

FC

Loss régression  
(L1, L2, etc.)

↑

Boîte cible :  
[x,y,w,h]

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, InceptionNet, etc  
(souvent préentraîné sur imageNet)

95

---

---

---

---

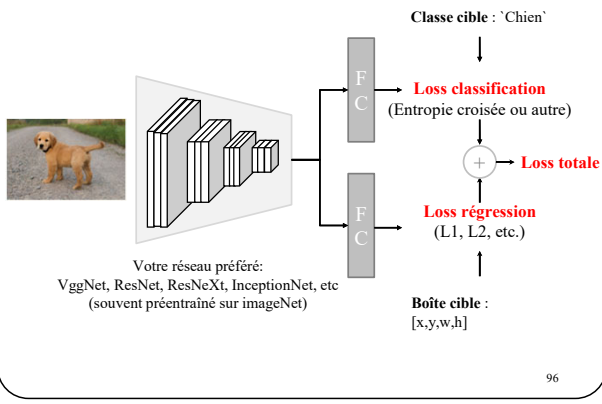
---

---

---

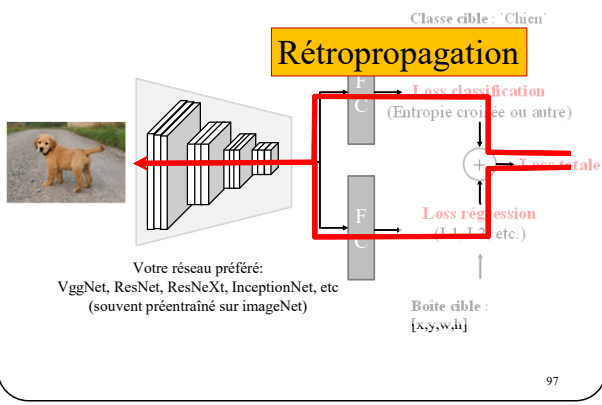
---

## Détection d'un seul objet



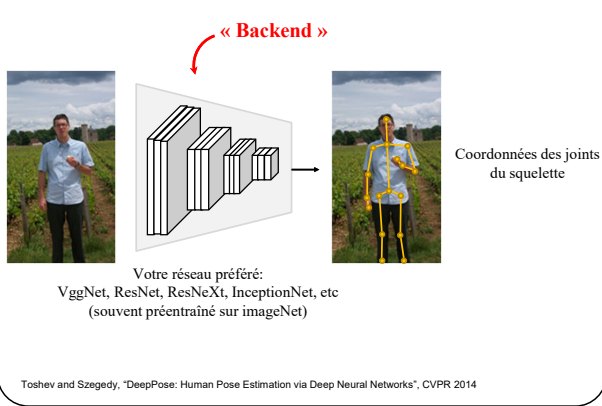
96

## Détection d'un seul objet



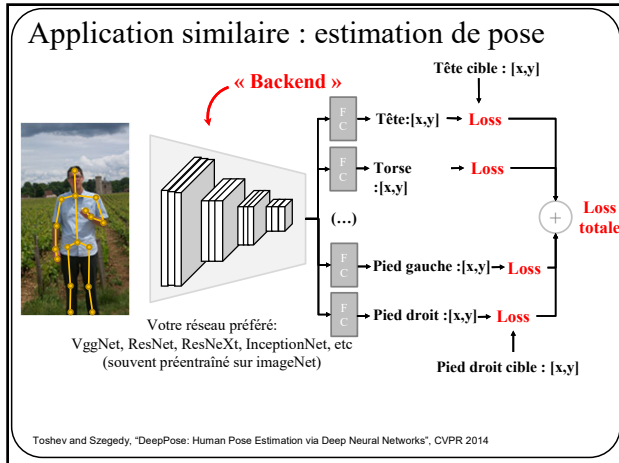
97

## Application similaire : estimation de pose

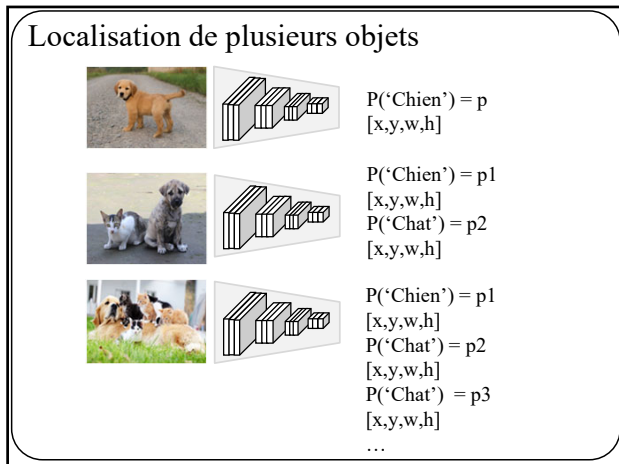


98

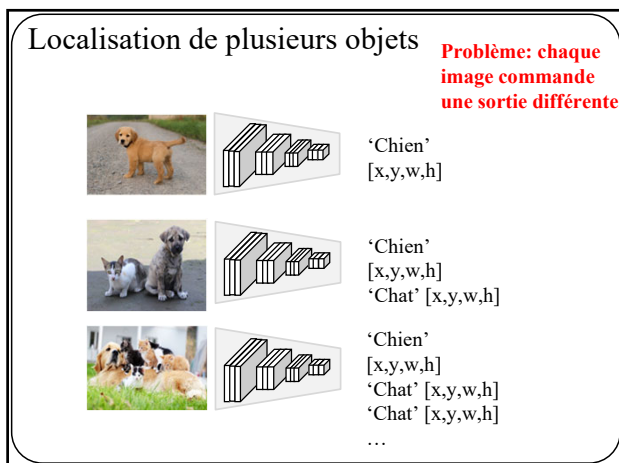




99



100

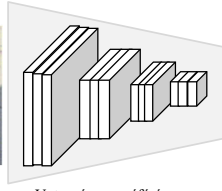


101

### Localisation de plusieurs objets

**Solution 1 :** appliquer un CNN à une fenêtre coulissante

3 classes : 'Chien', 'Chat', 'Fond'



**P('Fond')**  
**= 1**

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, etc  
(souvent préentraîné sur imageNet)

102

---

---

---

---

---

---

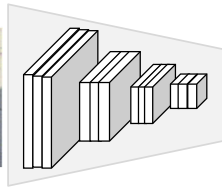
---

---

### Localisation de plusieurs objets

**Solution 1 :** appliquer un CNN à une fenêtre coulissante

3 classes : 'Chien', 'Chat', 'Fond'



**P('Fond')**  
**= 0.7**

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, etc  
(souvent préentraîné sur imageNet)

103

---

---

---

---

---

---

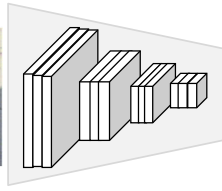
---

---

### Localisation de plusieurs objets

**Solution 1 :** appliquer un CNN à une fenêtre coulissante

3 classes : 'Chien', 'Chat', 'Fond'



**P('Chien')**  
**= 0.9**

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, etc  
(souvent préentraîné sur imageNet)

104

---

---

---

---

---

---

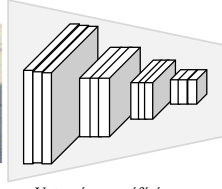
---

---

### Localisation de plusieurs objets

**Solution 1 :** appliquer un CNN à une fenêtre coulissante

3 classes : 'Chien', 'Chat', 'Fond'



**P('Fond')**  
**= 0.5**

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, etc  
(souvent préentraîné sur imageNet)

105

---

---

---

---

---

---

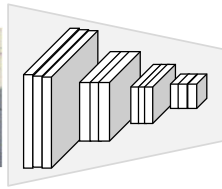
---

---

### Localisation de plusieurs objets

**Solution 1 :** appliquer un CNN à une fenêtre coulissante

3 classes : 'Chien', 'Chat', 'Fond'



**P('Chat')**  
**= 0.9**

Votre réseau préféré:  
VggNet, ResNet, ResNeXt, etc  
(souvent préentraîné sur imageNet)

106

---

---

---

---

---

---

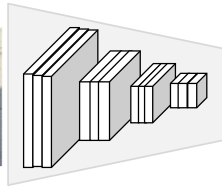
---

---

### Localisation de plusieurs objets

**Solution 1 :** appliquer un CNN à une fenêtre coulissante

3 classes : 'Chien', 'Chat', 'Fond'



**P('Chat')**  
**= 0.9**

L'inconvénient de cette méthode est qu'elle requière de traiter  
un très grand nombre de fenêtres coulissantes de plusieurs dimensions.

107

---

---

---

---

---

---

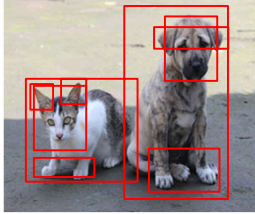
---

---

## Localisation de plusieurs objets

**Solution 2 :** Présélectionner un nombre restreint de fenêtres.

Il est relativement facile et rapide de trouver ~1000 fenêtres susceptibles de contenir un objet d'intérêt



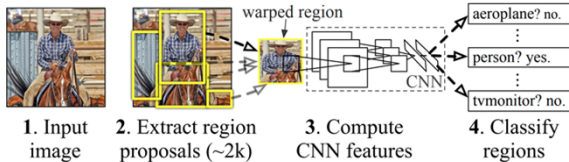
On appelle ce type de méthodes  
« *Region proposal method* »

Alene et al., "Measuring the objectness of image windows", TPAMI 2012  
Uijlings et al., "Selective Search for Object Recognition", IJCV 2013  
Cheng et al., "R-ING: Binarized normed gradients for objectness estimation at 300fps", CVPR 2014  
Zirnick and Dollar, "Edge boxes: Locating object proposals from edges", ECCV 2014

108

## R-CNN [Girshick et al, 2014]

### R-CNN: Regions with CNN features



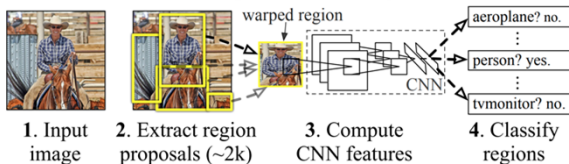
1. Extraire des régions (de 1000 à 2000)  
à l'aide d'une « *region proposal method* »

Girshick et al., "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014    109

109

## R-CNN [Girshick et al, 2014]

### R-CNN: Regions with CNN features

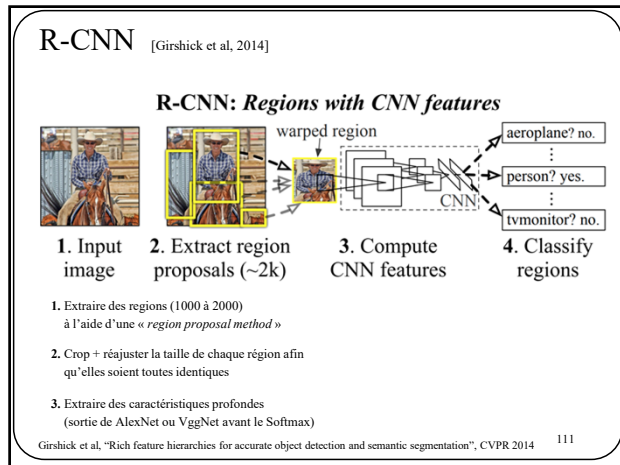


1. Extraire des régions (1000 à 2000)  
à l'aide d'une « *region proposal method* »

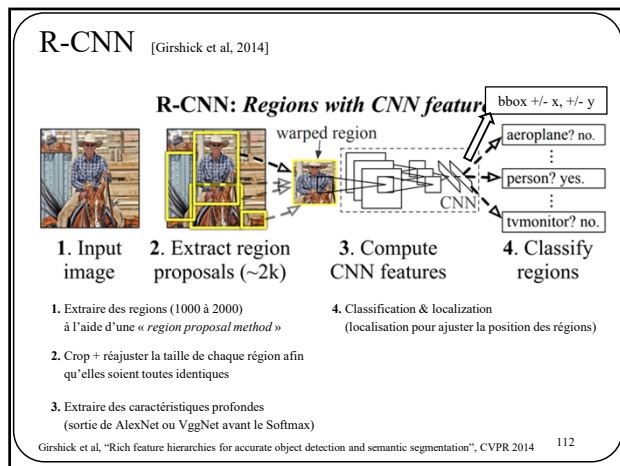
2. Crop + réajuster la taille de chaque région afin  
qu'elles soient toutes identiques

Girshick et al., "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014    110

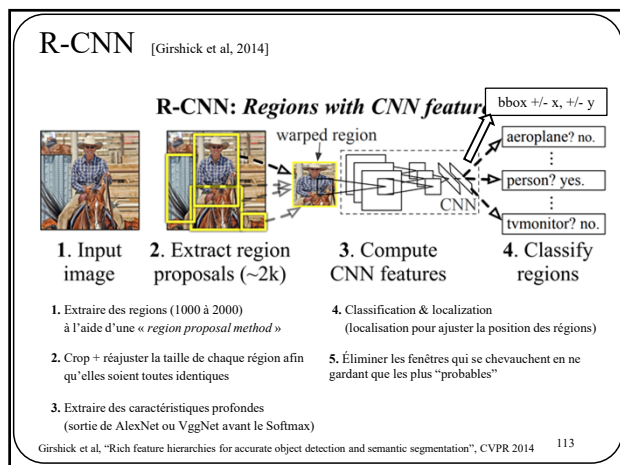
110



111



112



113

**R-CNN** [Girshick et al, 2014]

**R-CNN: Regions with CNN features**

Composantes:

- *Region-proposal method*
- "Backend" (AlexNet ou VGG16) pré-entraîné sur ImageNet puis *finetuned* sur Pascal VOC
- SVM *par classe* pour la classification
- Régresseur pour bouger les régions

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014 114

114

(Parenthèse)

Pour mieux comprendre

115

115

**Intersection over Union** Aussi appelé "Jaccard Index"

Comment comparer les régions ?

$$\frac{|Intersection|}{|Union|}$$

0.5 est "correct"  
0.7 est bon  
0.9 est excellent

116


116

**Intersection over Union** Aussi appelé "Jaccard Index"

Comment comparer les régions ?

$$\frac{|\text{Intersection}|}{|\text{Union}|}$$

0.5 est "correct"  
0.7 est bon  
0.9 est excellent



117

---

---

---

---

---

---

---


---

**Intersection over Union** Aussi appelé "Jaccard Index"

Comment comparer les régions ?

$$\frac{|\text{Intersection}|}{|\text{Union}|}$$

0.5 est "correct"  
0.7 est bon  
0.9 est excellent



118

---

---

---

---

---

---

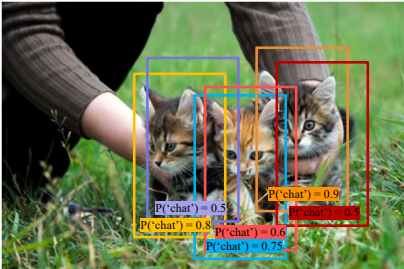
---

---

**"Non-Max Suppression"**

Comment éviter les duplicatas (2 boîtes sur un même objet) ?

- Sélectionner la boîte avec le plus haut score
- Éliminer les boîtes avec un  $\text{IoU} > \epsilon$  (p.e. 0.7) ayant un score moins élevé
- Répéter jusqu'à ce qu'aucune boîte ne puisse être éliminée



119

---

---

---

---

---

---

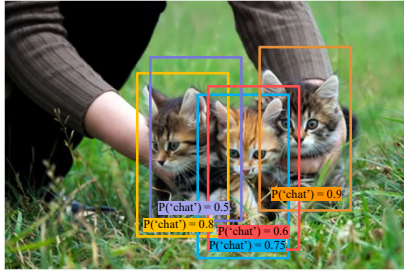
---

---

### “Non-Max Suppression”

Comment éviter les duplicatas (2 boîtes sur un même objet) ?

1. Sélectionner la boîte avec le plus haut score
2. Éliminer les boîtes avec un  $\text{IoU} > \epsilon$  (p.e. 0.7) ayant un score moins élevé
3. Répéter jusqu'à ce qu'aucune boîte ne puisse être éliminée



120

---

---

---

---

---

---

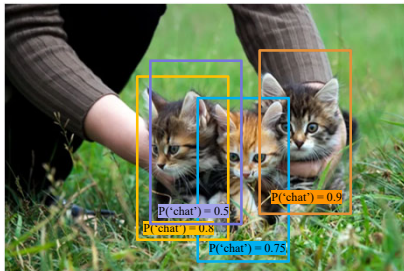
---

---

### “Non-Max Suppression”

Comment éviter les duplicatas (2 boîtes sur un même objet) ?

1. Sélectionner la boîte avec le plus haut score
2. Éliminer les boîtes avec un  $\text{IoU} > \epsilon$  (p.e. 0.7) ayant un score moins élevé
3. Répéter jusqu'à ce qu'aucune boîte ne puisse être éliminée



121

---

---

---

---

---

---

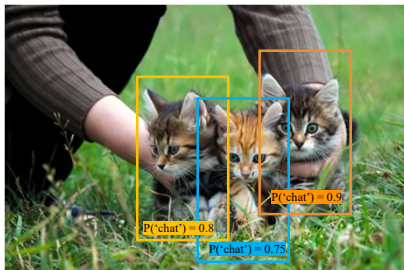
---

---

### “Non-Max Suppression”

Comment éviter les duplicatas (2 boîtes sur un même objet) ?

1. Sélectionner la boîte avec le plus haut score
2. Éliminer les boîtes avec un  $\text{IoU} > \epsilon$  (p.e. 0.7) ayant un score moins élevé
3. Répéter jusqu'à ce qu'aucune boîte ne puisse être éliminée



122

---

---

---

---

---

---

---

---



**“mean Average Precision”**

Comment mesurer la performance d’un détecteur d’objet ?

Doit évaluer la qualité de la prédiction ET de la localisation

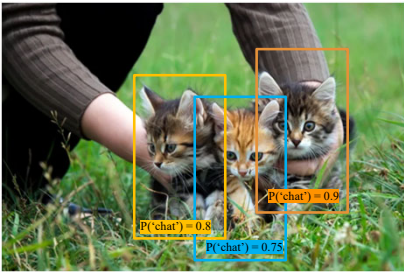
Classification:

- “Top 1%”
- “Top 5%”
- Top ..%

Segmentation:

- (Sorensen-)Dice/F1
- IoU/Jaccard Index
- Précision

Localisation ?



123

---

---

---

---

---

---

---

---

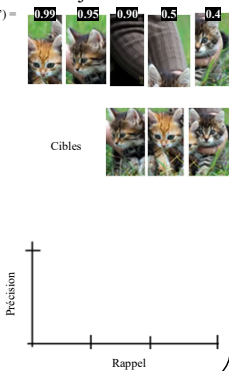
**“mean Average Precision”**

Comment mesurer la performance d’un détecteur d’objet ?

Doit évaluer la qualité de la prédiction ET de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par probabilité
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe
3. Faire la moyenne pour chaque image
4. Faire la moyenne pour toutes les classes

Adapté de [https://web.eecs.umich.edu/~junting/videos/eecs498/498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~junting/videos/eecs498/498_FA2019_lecture15.pdf)



124

---

---

---

---

---

---

---

---

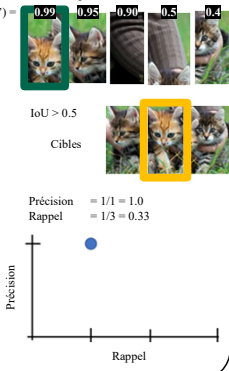
**“mean Average Precision”**

Comment mesurer la performance d’un détecteur d’objet ?

Doit évaluer la qualité de la prédiction ET de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par probabilité
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe
3. Faire la moyenne pour chaque image
4. Faire la moyenne pour toutes les classes

Adapté de [https://web.eecs.umich.edu/~junting/videos/eecs498/498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~junting/videos/eecs498/498_FA2019_lecture15.pdf)



125

---

---

---

---

---

---

---

---


**“mean Average Precision”**

Comment mesurer la performance d’un détecteur d’objet ?


Doit évaluer la qualité de la prédiction *ET* de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par probabilité
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe
3. Faire la moyenne pour chaque image
4. Faire la moyenne pour toutes les classes

$P('cat') =$

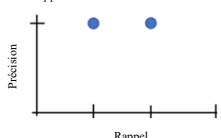


$IoU > 0.5$



Cibles

Précision =  $2/2 = 1.0$   
Rappel =  $2/3 = 0.67$



Adapté de [https://web.eecs.umich.edu/~junting/videos/eecs498-498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~junting/videos/eecs498-498_FA2019_lecture15.pdf)

126


**“mean Average Precision”**

Comment mesurer la performance d’un détecteur d’objet ?


Doit évaluer la qualité de la prédiction *ET* de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par probabilité
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe
3. Faire la moyenne pour chaque image
4. Faire la moyenne pour toutes les classes

$P('cat') =$

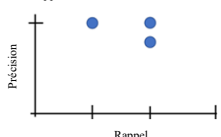


$IoU \leq 0.5$



Cibles

Précision =  $2/3 = 0.67$   
Rappel =  $2/3 = 0.67$



Adapté de [https://web.eecs.umich.edu/~junting/videos/eecs498-498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~junting/videos/eecs498-498_FA2019_lecture15.pdf)

127


**“mean Average Precision”**

Comment mesurer la performance d’un détecteur d’objet ?


Doit évaluer la qualité de la prédiction *ET* de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par probabilité
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe
3. Faire la moyenne pour chaque image
4. Faire la moyenne pour toutes les classes

$P('cat') =$

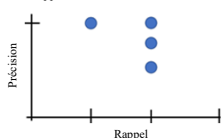


$IoU \leq 0.5$



Cibles

Précision =  $2/4 = 0.5$   
Rappel =  $2/3 = 0.67$



Adapté de [https://web.eecs.umich.edu/~junting/videos/eecs498-498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~junting/videos/eecs498-498_FA2019_lecture15.pdf)

128

### “mean Average Precision”

Comment mesurer la performance d’un détecteur d’objet ?

Doit évaluer la qualité de la prédiction *ET* de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par probabilité
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe

- 3. Faire la moyenne pour chaque image
- 4. Faire la moyenne pour toutes les classes

Adapté de [https://web.eecs.umich.edu/~jursky/videos/eecs498-498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~jursky/videos/eecs498-498_FA2019_lecture15.pdf)

129

### “mean Average Precision”

Comment mesurer la performance d’un détecteur d’objet ?

Doit évaluer la qualité de la prédiction *ET* de la localisation

1. Faire la détection sur toutes les images de test
2. Pour chaque image et chaque classe
  - a. Pour chaque détection
    - i. Ordonner les détections par
    - ii. Prendre la détection la plus probable
      1. Si elle correspond à une cible, l’indiquer comme vrai positif et éliminer la cible correspondante
      2. Sinon, l’indiquer comme faux positif
    - iii. Tracer le point
  - b. Calculer l’aire sous la courbe  
=> *average precision*
3. Faire la moyenne pour chaque image
4. Faire la moyenne pour toutes les classes => **mAP**

Adapté de [https://web.eecs.umich.edu/~jursky/videos/eecs498-498\\_FA2019\\_lecture15.pdf](https://web.eecs.umich.edu/~jursky/videos/eecs498-498_FA2019_lecture15.pdf)

130

### “mean Average Precision”

Comment mesurer la performance d’un détecteur d’objet ?

Exemples:

SSD MobileNet Performance: mAP=0.34

Precision-Recall curve for Person

Van Erten, A. (2019, January). Satellite imagery multiscale rapid detection with windowed networks. In 2019 IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 735-743). IEEE.


<https://medium.com/@timothycarlton/understanding-the-map-evaluation-metric-for-object-detection-a0766962c4f3>

131

Jeux de données populaires


Pascal VOC (2005-2012)

Classification/Detection (20 classes)



http://host.robots.ox.ac.uk/pascal/VOC/voc2012/index.html

Action Classification (10 classes + "other")



132

---

---

---

---

---

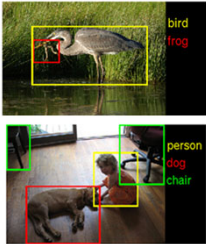
---

---

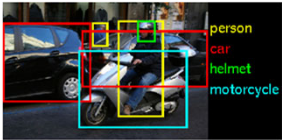
---

Jeux de données populaires

ImageNet (2013+)



https://image-net.org/challenges/LSVRC/2013/



Comparative scale

	PASCAL VOC 2012	ILSVRC 2013
Number of object classes	20	200
Training		
Num images	5717	395909
Num objects	13609	345854
Validation		
Num images	5823	20121
Num objects	13841	55502
Testing		
Num images	10991	40152
Num objects	---	---

133

---

---

---

---

---

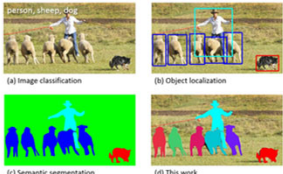
---

---

---

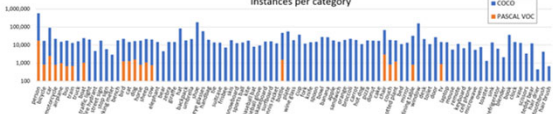
Jeux de données populaires

MS-COCO (2014+) 100 classes



(a) Image classification (b) Object localization (c) Semantic segmentation (d) This work

Instances per category



Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

134

---

---

---

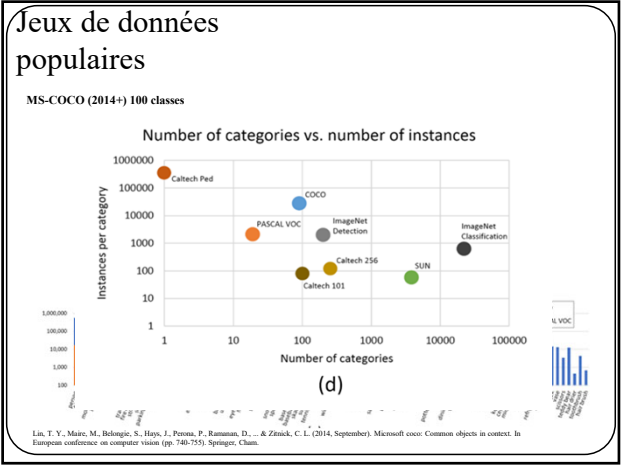
---

---

---

---

---



135

---

---

---

---

---

---

---

---



136

---

---

---

---

---

---

---

---

### R-CNN [Girshick et al, 2014]

**R-CNN: Regions with CNN features**

1. Input image    2. Extract region proposals (~2k)    3. Compute CNN features    4. Classify regions

bbox +/- x, +/- y

aeroplane? no.

person? yes.

tvmonitor? no.

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM-v5 [20]	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [19]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [41]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [18]	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	71.8	65.8	53.0	36.8	35.9	59.7	60.0	69.9	27.9	50.6	41.4	70.0	62.0	69.0	58.1	29.5	59.4	39.3	61.2	52.4	53.7

Table 1: Detection average precision (%) on VOC 2010 test. R-CNN is most directly comparable to UVA and Regionlets since all methods use selective search region proposals. Bounding-box regression (BB) is described in Section C. At publication time, SegDPM was the top-performer on the PASCAL VOC leaderboard. <sup>1</sup>DPM and SegDPM use context rescoring not used by the other methods.

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014    137

137

---

---

---

---

---

---

---

---

## R-CNN [Girshick et al, 2014]

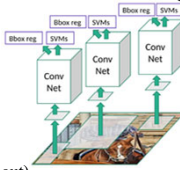
**Problème du R-CNN**

3 entraînements séparés (pas d'entraînement bout-en-bout)

- *Finetuning* du CNN (entropie croisée)
  - pré-entraîné sur ImageNet
  - ré-entraîné sur Pascal VOC
- Entraînement du SVM (Hinge loss)
- Entraînement de la régression (loss L2)

Entraînement lent et complexe

- 84h
- Détection lente
- 47secondes / image avec VGG16



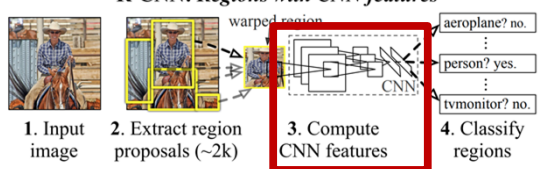
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014 138

138

## R-CNN [Girshick et al, 2014]

**Problème du R-CNN**

**R-CNN: Regions with CNN features**



1. Input image    2. Extract region proposals (~2k)    3. Compute CNN features    4. Classify regions

2000 propagations avant par image !

**Training time (Hours)**

Model	Training time (Hours)
R-CNN	84

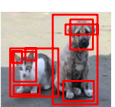
**Test time (seconds)**

Model	Test time (seconds)
R-CNN	49

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014 139

139

## Fast R-CNN [Girshick, 2015]



640x480x3

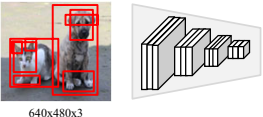
1. Localiser des régions

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

140

### Fast R-CNN [Girshick, 2015]

« Backend »



640x480x3

1. Localiser des régions
2. Propagation avant de l'image dans 5 blocs convolutionnels

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

141

---

---

---

---

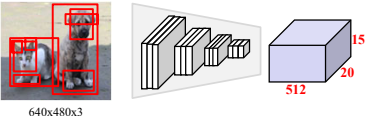
---

---

---

---

### Fast R-CNN [Girshick, 2015]



640x480x3

1. Localiser des régions
2. Propagation avant de l'image dans 5 blocs convolutionnels.  
Cartes d'activation **20x15x512**

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

142

---

---

---

---

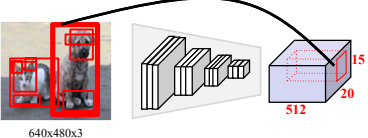
---

---

---

---

### Fast R-CNN [Girshick, 2015]



640x480x3

1. Localiser des régions
2. Propagation avant de l'image dans 5 blocs convolutionnels.  
Cartes d'activation **20x15x512**
3. À tour de rôle, projeter chaque région vers les cartes d'activation.

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

143

---

---

---

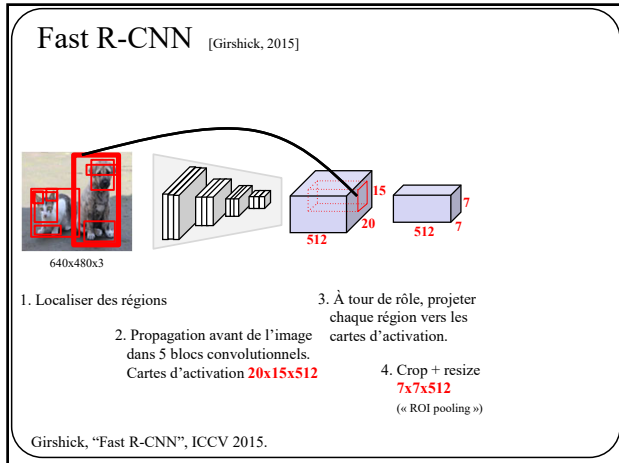
---

---

---

---

---



144

---

---

---

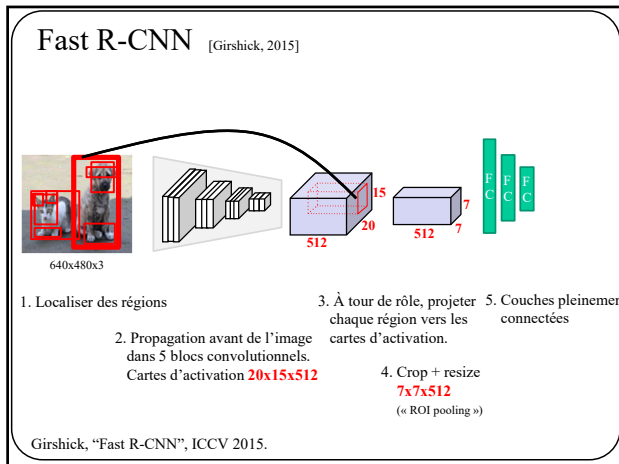
---

---

---

---

---



145

---

---

---

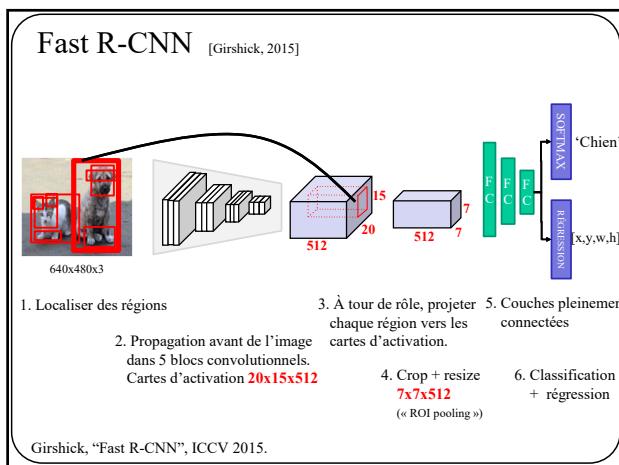
---

---

---

---

---



146

---

---

---

---

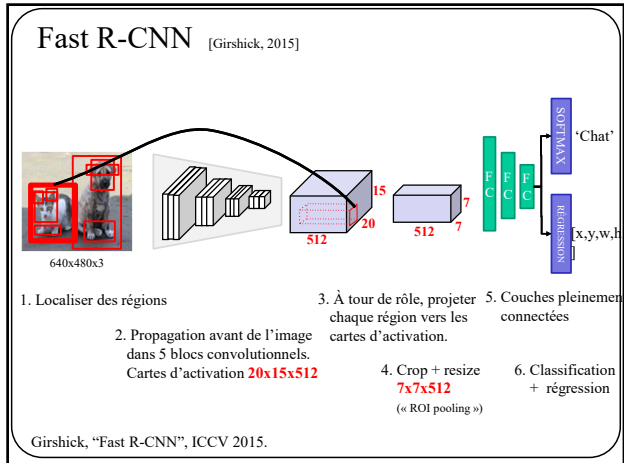
---

---

---

---





147

---

---

---

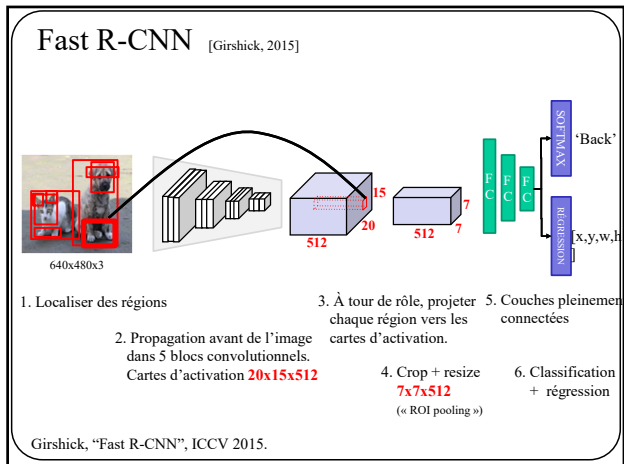
---

---

---

---

---



148

---

---

---

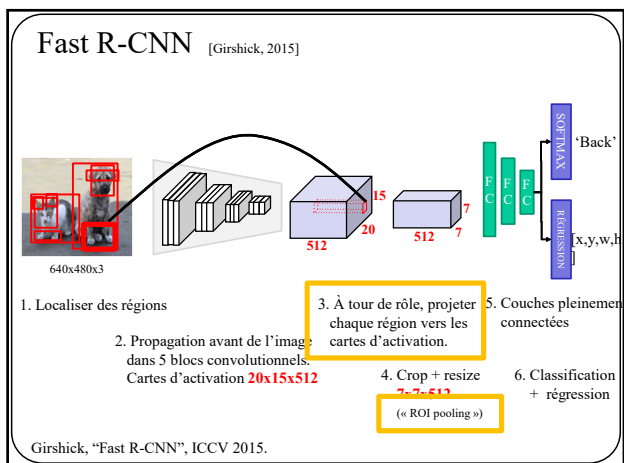
---

---

---

---

---



149

---

---

---

---

---

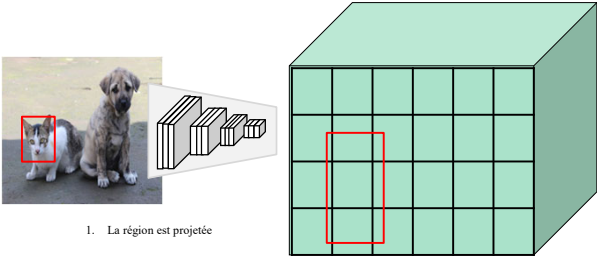
---

---

---

### ROI pooling

Comment transformer une région de taille arbitraire en taille fixe pour un CNN ?



1. La région est projetée

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

150

---

---

---

---

---

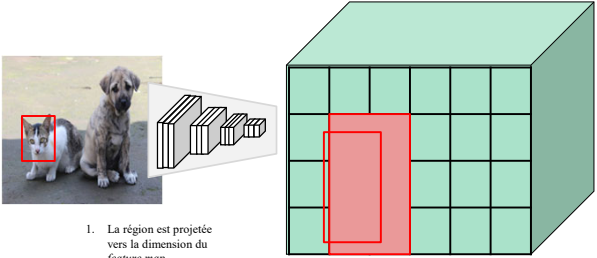
---

---

---

### ROI pooling

Comment transformer une région de taille arbitraire en taille fixe pour un CNN ?



1. La région est projetée vers la dimension du *feature map*

2. La région est ajustée à la grille (*snap*)

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

151

---

---

---

---

---

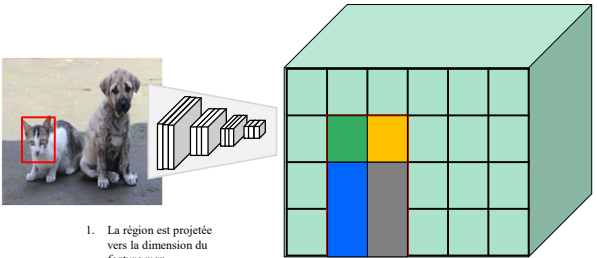
---

---

---

### ROI pooling

Comment transformer une région de taille arbitraire en taille fixe pour un CNN ?



1. La région est projetée vers la dimension du *feature map*

2. La région est ajustée à la grille (*snap*)

3. La grille est subdivisée en  $n \times n$  régions cibles (p.e.  $7 \times 7$ )

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

152

---

---

---

---

---

---

---

---

## ROI pooling

Comment transformer une région de taille arbitraire en taille fixe pour un CNN ?

1. La région est projetée vers la dimension du *feature map*
2. La région est ajustée à la grille (*snip*)
3. La grille est subdivisée en  $n \times n$  régions cibles (p.e.  $7 \times 7$ )
4. max pooling

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

153

## Fast R-CNN [Girshick, 2015]

Autre illustration (de Girshick):

**Avantages:**

- 1 propagation avant par image au lieu de 1 par région
- Entraînement bout-en-bout

**Inconvénients:**

- La "region proposal method" indépendante du réseau

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

154

## Fast R-CNN [Girshick et al, 2015]

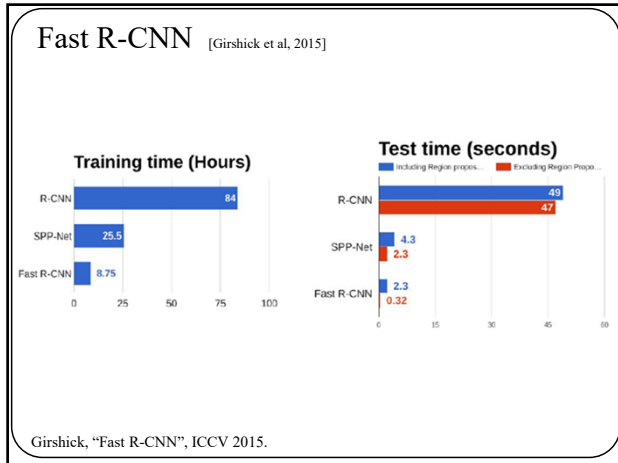
Tiré de l'article

method	train set	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
BabyLearning	Prop.	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.8	77.9	64.0	35.3	67.9	55.7	68.7	62.6	65.2
NUS,NIN,c2000	Unk.	80.2	73.8	61.9	43.7	43.8	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3	63.8
R-CNN BB [10]	12	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3	62.4
FRCN [ours]	12	80.3	74.7	66.9	46.9	37.7	73.9	68.6	87.7	41.7	71.1	51.1	86.0	77.8	79.8	69.8	32.1	65.5	63.8	76.4	61.7	65.7
FRCN [ours]	07+12	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.8	80.8	80.8	72.0	35.1	68.3	65.7	80.4	64.2	68.4

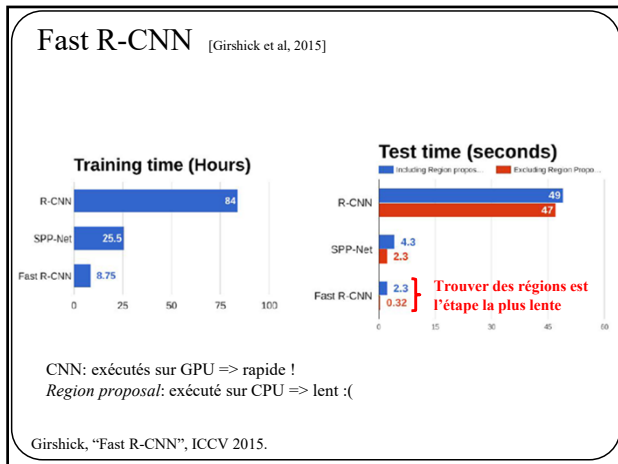
Table 3. VOC 2012 test detection average precision (%). BabyLearning and NUS,NIN,c2000 use networks based on [17]. All other methods use VGG16. Training set key: see Table 2. Unk.: unknown.

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

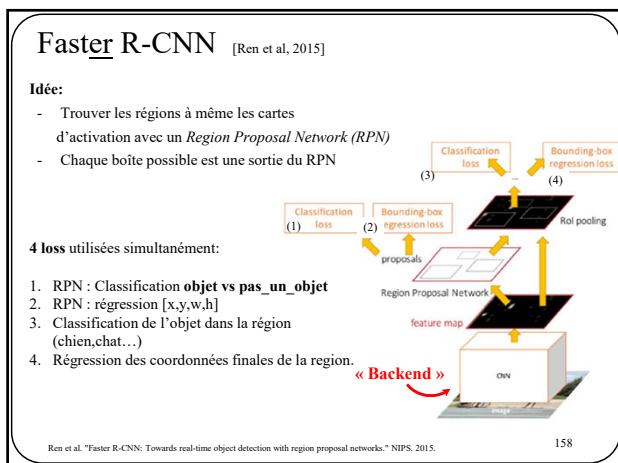
155



156



157



158

classifier

RoI pooling

proposals

Region Proposal Network

feature maps

« Backend »

conv layers

Deux étapes:

Classifier les régions

Proposer les régions

Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." NIPS, 2015.

159

159

---

---

---

---

---

---

---

---

classifier

RoI pooling

proposals

Region Proposal Network

feature maps

« Backend »

conv layers

Utilisation de "anchors"

Boîtes de forme prédéterminée

En pratique:  $k = 9$

2k scores

4k coordinates

cls layer

reg layer

256-d

intermediate layer

sliding window

conv feature map

4 anchor boxes

Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." NIPS, 2015.

160

160

---

---

---

---

---

---

---

---

classifier

RoI pooling

proposals

Region Proposal Network

feature maps

« Backend »

conv layers

R-CNN Test-Time Speed

R-CNN

49

SPP-Net

4.3

Fast R-CNN

2.3

Faster R-CNN

0.2

Table 5: Timing (ms) on a K40 GPU, except SS proposal is evaluated in a CPU. "Region-wise" includes NMS, pooling, fully-connected, and softmax layers. See our released code for the profiling of running time.

model	system	conv	proposal	region-wise	total	rate
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps
VGG	RPN + Fast R-CNN	141	10	47	198	5 fps
ZF	RPN + Fast R-CNN	31	3	25	59	17 fps

Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." NIPS, 2015.

161

161

---

---

---

---

---

---

---

---


53




YOLO

[Redmon2017, Redmon2018]

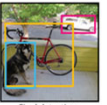
Ainsi, YOLO prédit TOUJOURS 49x2=98 objets possibles chacune avec un indice de confiance.



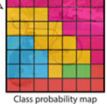
5 x 5 grid on input



Bounding boxes + confidence



Final detections



Class probability map

J. Redmon, S Divvala, R Girshick, A Farhad "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2017

J. Redmon, A. Farhadi "YOLO9000: Better, Faster, Stronger", CVPR 2018

165

165

YOLO

[Redmon2017, Redmon2018]

	mAP	acc	bike	bird	boat	botle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
SR-CNN-MORE-DATA [1]	73.9	83.5	82.9	76.4	59.8	62.7	79.4	77.2	80.6	55.0	79.1	62.2	87.0	83.4	84.7	79.3	45.3	71.4	65.5	80.3	74.0
HyperNet-VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7
HyperNet-SP	71.3	84.2	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	80.4	78.5	73.5	55.8	43.4	79.1	73.1	89.4	49.4	75.5	57.0	87.5	80.9	81.0	74.7	41.8	71.5	68.5	82.1	67.2
MR-CNN-S-CNN [1]	70.7	83.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Fast R-CNN [1]	70.4	84.9	79.8	74.1	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEPLENS-COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	68.8	73.9	71.4
NoC [2]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [1]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH-FGS-STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS-NK-C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	68.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabylLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS-NK	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [13]	56.3	74.8	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	50.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.0	61.2	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [10]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [1]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	28.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

Table 3: PASCAL VOC 2012 Leaderboard. YOLO compared with the full comp4 (outside data allowed) public leaderboard as of November 6th, 2015. Mean average precision and per-class average precision are shown for a variety of detection methods. YOLO is the only real-time detector. Fast R-CNN + YOLO is the forth highest scoring method, with a 2.3% boost over Fast R-CNN.

J. Redmon, S Divvala, R Girshick, A Farhad "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2017

J. Redmon, A. Farhadi "YOLO9000: Better, Faster, Stronger", CVPR 2018

166

166

YOLO

[Redmon2017, Redmon2018]

	mAP	acc	bike	bird	boat	botle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
SR-CNN-MORE-DATA [1]	73.9	83.5	82.9	76.4	59.8	62.7	79.4	77.2	80.6	55.0	79.1	62.2	87.0	83.4	84.7	79.3	45.3	71.4	65.5	80.3	74.0
HyperNet-VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7
HyperNet-SP	71.3	84.2	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	80.4	78.5	73.5	55.8	43.4	79.1	73.1	89.4	49.4	75.5	57.0	87.5	80.9	81.0	74.7	41.8	71.5	68.5	82.1	67.2
MR-CNN-S-CNN [1]	70.7	83.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Fast R-CNN [1]	70.4	84.9	79.8	74.1	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEPLENS-COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	68.8	73.9	71.4
NoC [2]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [1]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH-FGS-STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS-NK-C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	68.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabylLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS-NK	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [13]	56.3	74.8	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	50.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.0	61.2	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [10]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [1]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	28.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

Ensemble de Fast R-CNN + YOLO

Moins bon que les autres, mais plus simple et rapide

Table 3: PASCAL VOC 2012 Leaderboard. YOLO compared with the full comp4 (outside data allowed) public leaderboard as of November 6th, 2015. Mean average precision and per-class average precision are shown for a variety of detection methods. YOLO is the only real-time detector. Fast R-CNN + YOLO is the forth highest scoring method, with a 2.3% boost over Fast R-CNN.

J. Redmon, S Divvala, R Girshick, A Farhad "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2017

J. Redmon, A. Farhadi "YOLO9000: Better, Faster, Stronger", CVPR 2018

167

167

55

## YOLOv2-3

v2:

- Batch norm
- Images de plus haute résolution
- *Anchors*
- et autres

v3:

- Réseau plus profond
- Détection à plusieurs résolutions
- Plus de boîtes par élément de grille

J. Redmon, S. Divvala, R. Girshick, A. Farhad "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2017  
J. Redmon, A. Farhadi "YOLO9000: Better, Faster, Stronger", CVPR 2018  
Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

168

168

## YOLOv2-v3

Yolo v2 et James Bond

<https://www.youtube.com/watch?v=VOC3huqHrss>

<https://pjreddie.com>  
(Site web du premier auteur)

J. Redmon, S. Divvala, R. Girshick, A. Farhad "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2017  
J. Redmon, A. Farhadi "YOLO9000: Better, Faster, Stronger", CVPR 2018  
J. Redmon, & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

169

169

## YOLOv2-v3

À ce jour,  
nous sommes  
à YOLOv7!

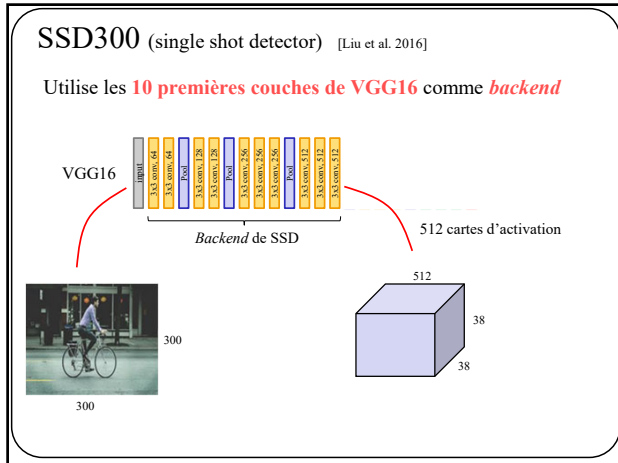
J. Redmon, S. Divvala, R. Girshick, A. Farhad "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2017  
J. Redmon, A. Farhadi "YOLO9000: Better, Faster, Stronger", CVPR 2018  
J. Redmon, & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

170

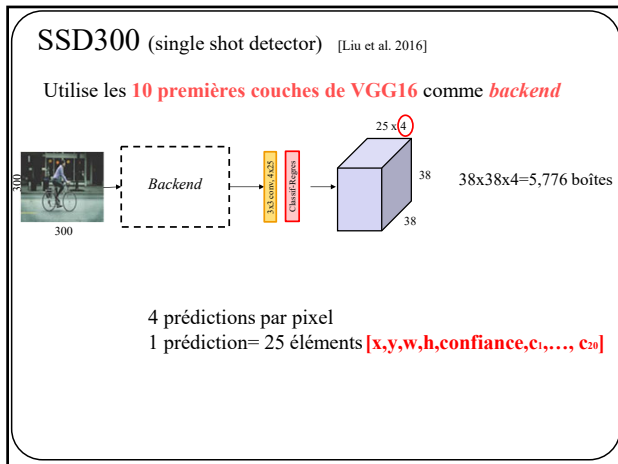
170



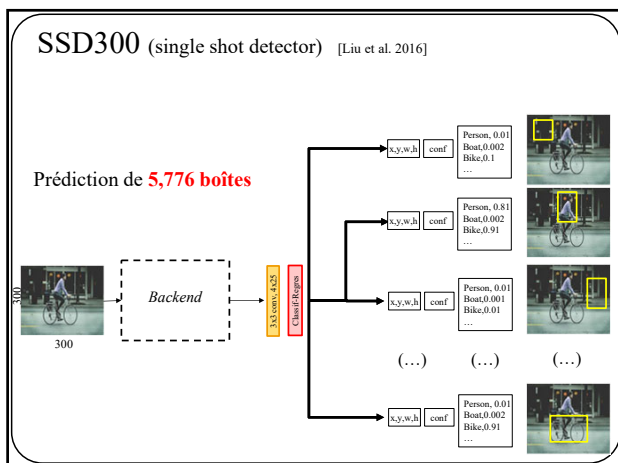




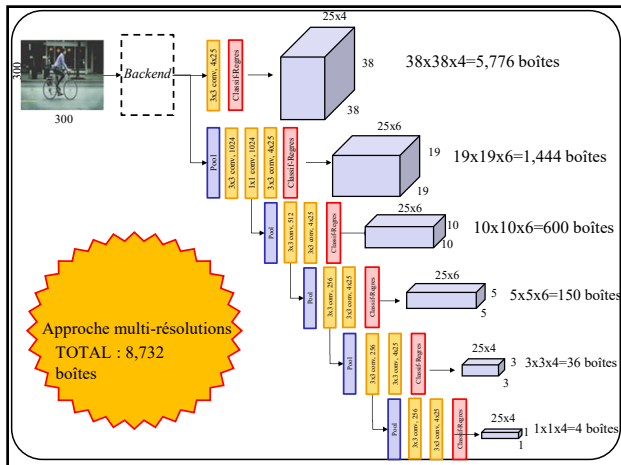
174



175



176



177

---

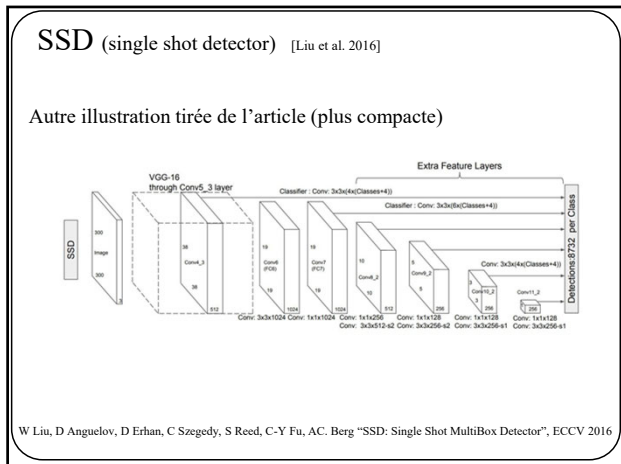
---

---

---

---

---



178

---

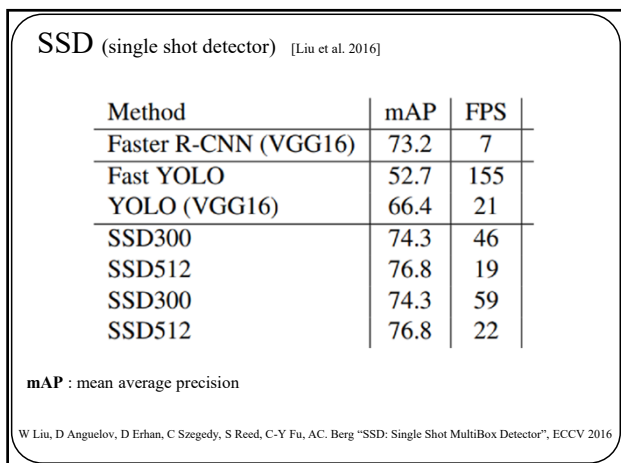
---

---

---

---

---



179

---

---

---

---

---

---

SSD (single shot detector) [Liu et al. 2016]

[https://medium.com/@jonathan\\_hui/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06](https://medium.com/@jonathan_hui/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06)



Excellent document sur SSD

180

---

---

---

---

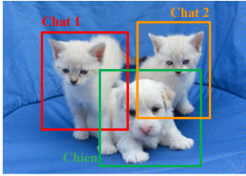
---

---


---

---

### Segmentation par instance



Localisation



Segmentation par instance

181

181

---

---

---

---

---


---

---

---

### Mask R-CNN [He et al., 2017]

Idée de base:



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

182

---

---

---

---

---

---

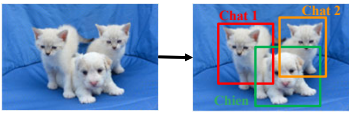
---

---

### Mask R-CNN [He et al., 2017]

Idée de base:

1. Localisation



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

183

---

---

---

---

---

---

---

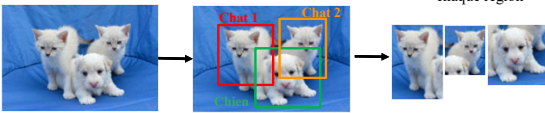
---

### Mask R-CNN [He et al., 2017]

Idée de base:

1. Localisation

2. Rogner (*crop*) chaque région



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

184

---

---

---

---

---

---

---

---

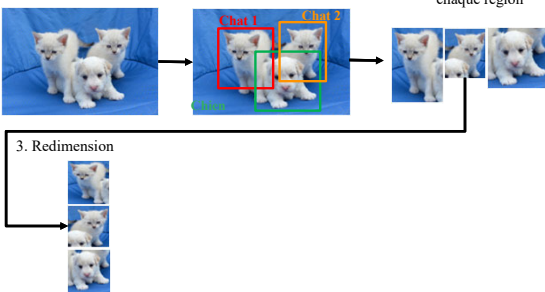
### Mask R-CNN [He et al., 2017]

Idée de base:

1. Localisation

2. Rogner (*crop*) chaque région

3. Redimension



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

185

---

---

---

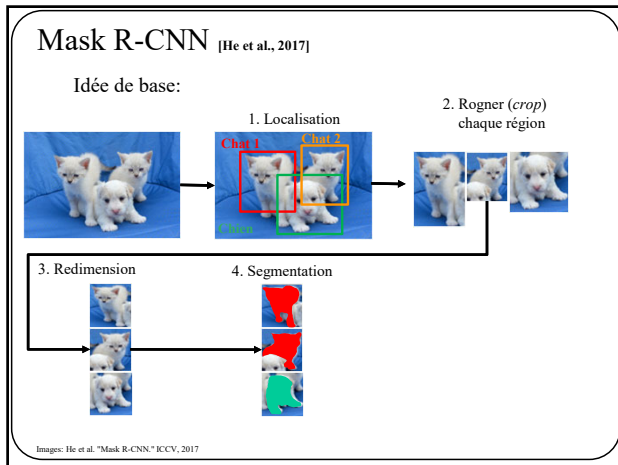
---

---

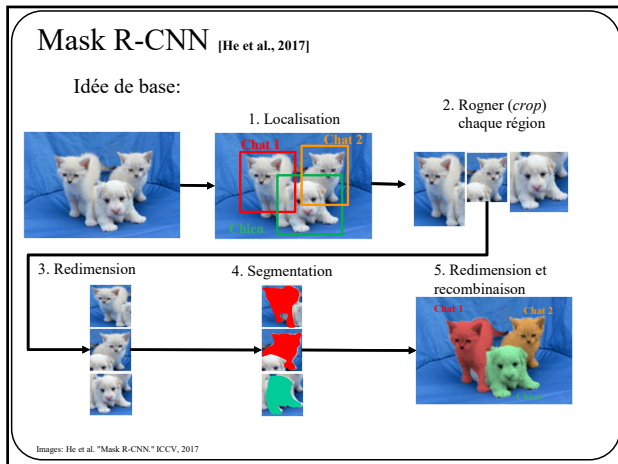
---

---

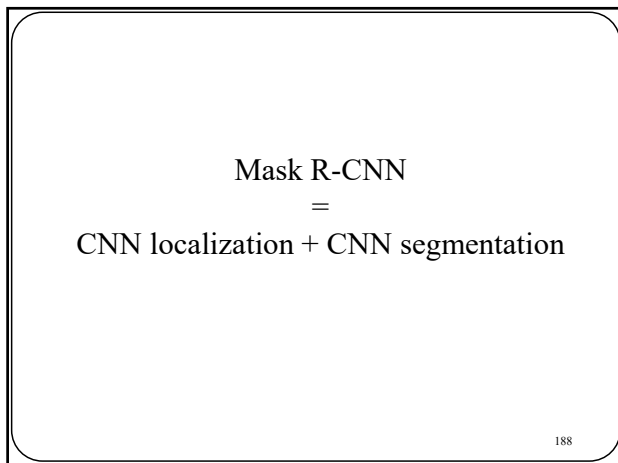
---



186



187



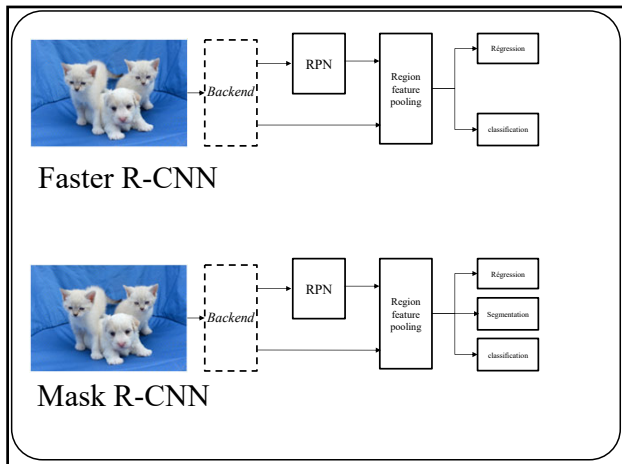
188

Mask R-CNN  
=  
Faster R-CNN (avec backend ResNet) + CNN segmentation

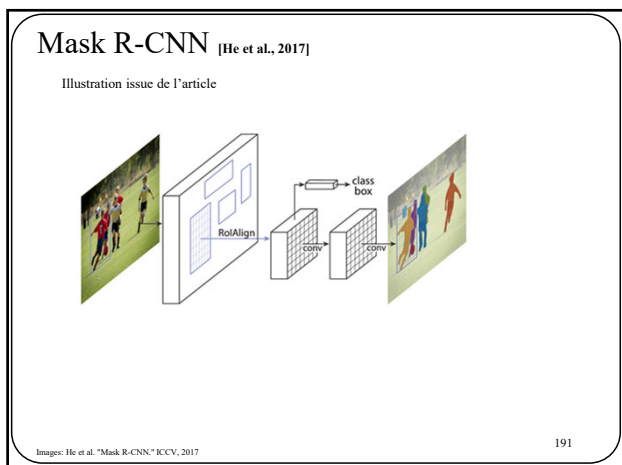
(Config de l'article d'origine. D'autres versions de Mask R-CNN utilisent d'autres backends.)

189

189



190



191





**Mask R-CNN** [He et al., 2017]

<https://www.analyticsvidhya.com/blog/2019/07/computer-vision-implementing-mask-r-cnn-image-segmentation/>

Excellent document sur **Mask R-CNN**

195

**En résumé**

**Bons survols:**

- Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., & Qu, R. (2019). A survey of deep learning-based object detection. IEEE access, 7, 128837-128868.
- Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., ... & Murphy, K. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7310-7311).

Leaderboard Dataset

<https://paperswithcode.com/sota/object-detection-on-coco>

196

196

**En résumé**

La détection d'objets et segmentation d'instances est complexe avec beaucoup d'astuces pour entraîner et prédire.

Deux grandes familles de méthodes:

**Two-stage detectors (e.g. fasterRCNN)**

- Plus complexes
- Plus lents
- Plus performants

**Single Stage detectors (e.g. yolo, SSD)**

- Plus simples
- Plus rapides
- Moins performants

Bibliothèque de détection d'objets/segmentation en Pytorch  
<https://github.com/facebookresearch/detectron2>

Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., & Qu, R. (2019). A survey of deep learning-based object detection. IEEE access, 7, 128837-128868.

197

197