

# Réseaux de neurones convolutifs en vision par ordinateur

## Apprentissage Profond

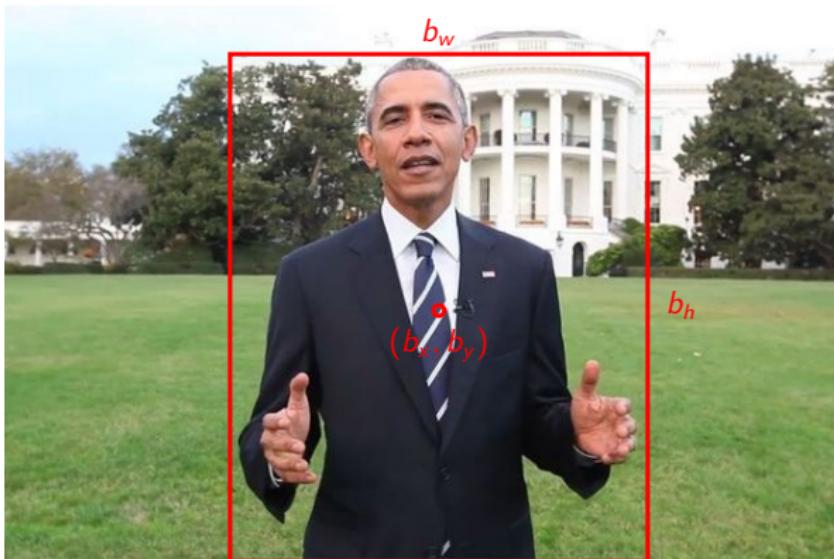
A. Carlier

2023

# Plan du cours

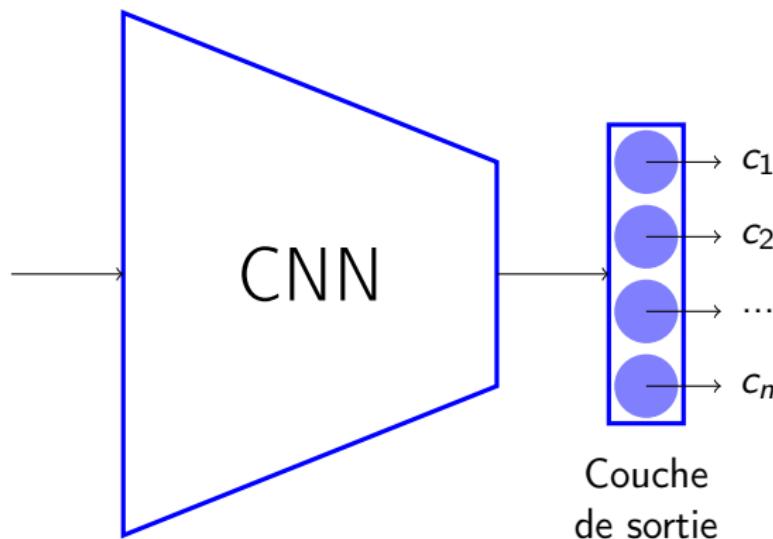
- 1 Localisation
- 2 Détection d'objet
- 3 Segmentation d'image
- 4 Estimation de pose
- 5 Reconnaissance de visage

# Classification et localisation conjointes



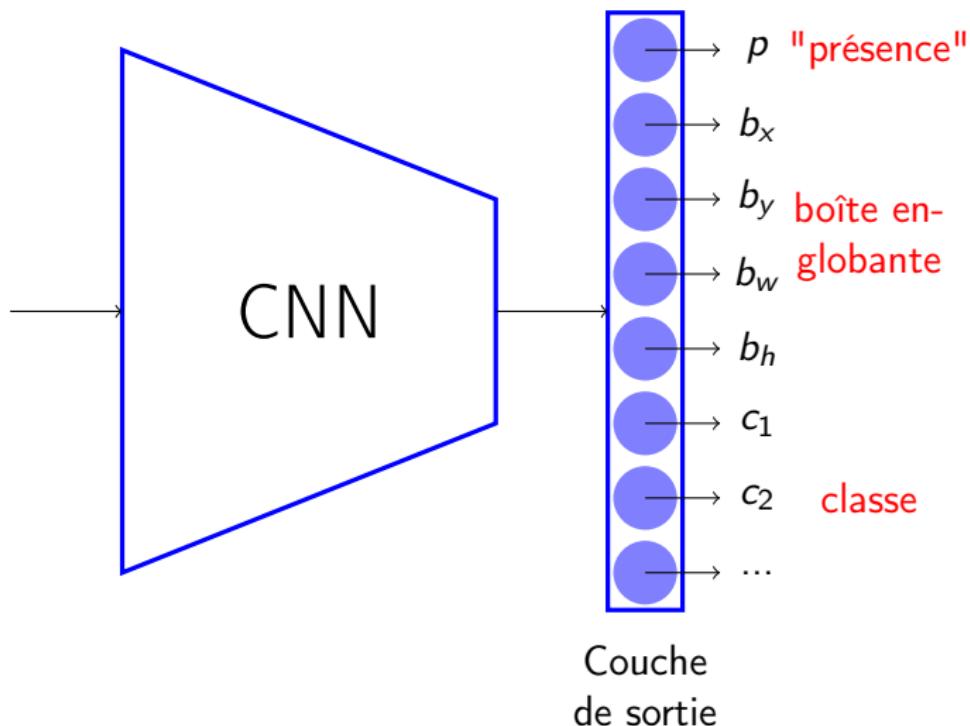
On cherche dans ce problème à classifier et localiser **un objet** par image.

# Classification et localisation conjointes



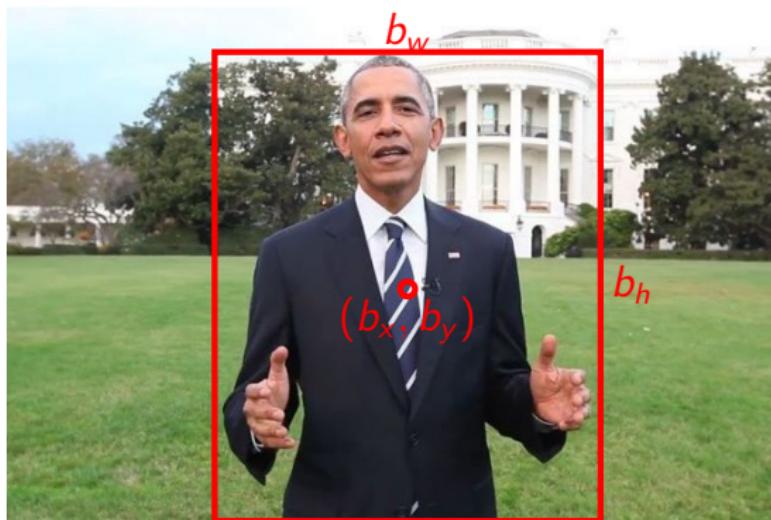
Nous devons ajouter une information spatiale sur l'objet classifié.

# Classification et localisation conjointes



Nous pouvons rajouter les coordonnées de la boîte englobante autour de l'objet.

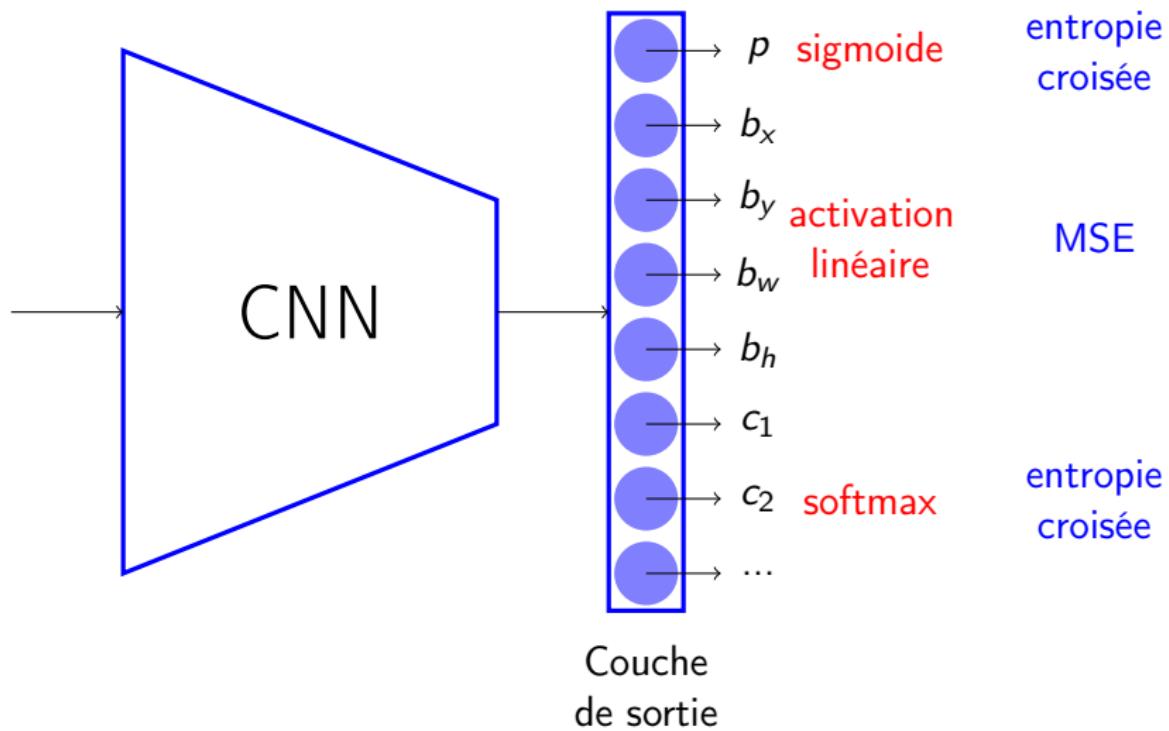
# Classification et localisation conjointes



$$\hat{y} = \begin{pmatrix} 1 \\ 0.52 \\ 0.46 \\ 0.5 \\ 0.91 \\ 0 \\ \dots \\ 1 \\ \dots \\ 0 \end{pmatrix}$$

# Classification et localisation conjointes

Quelle fonction de coût ?

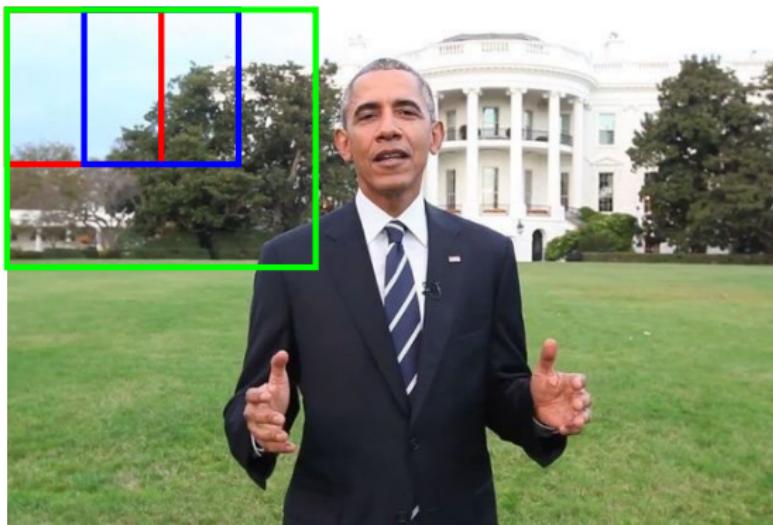


# Plan du cours

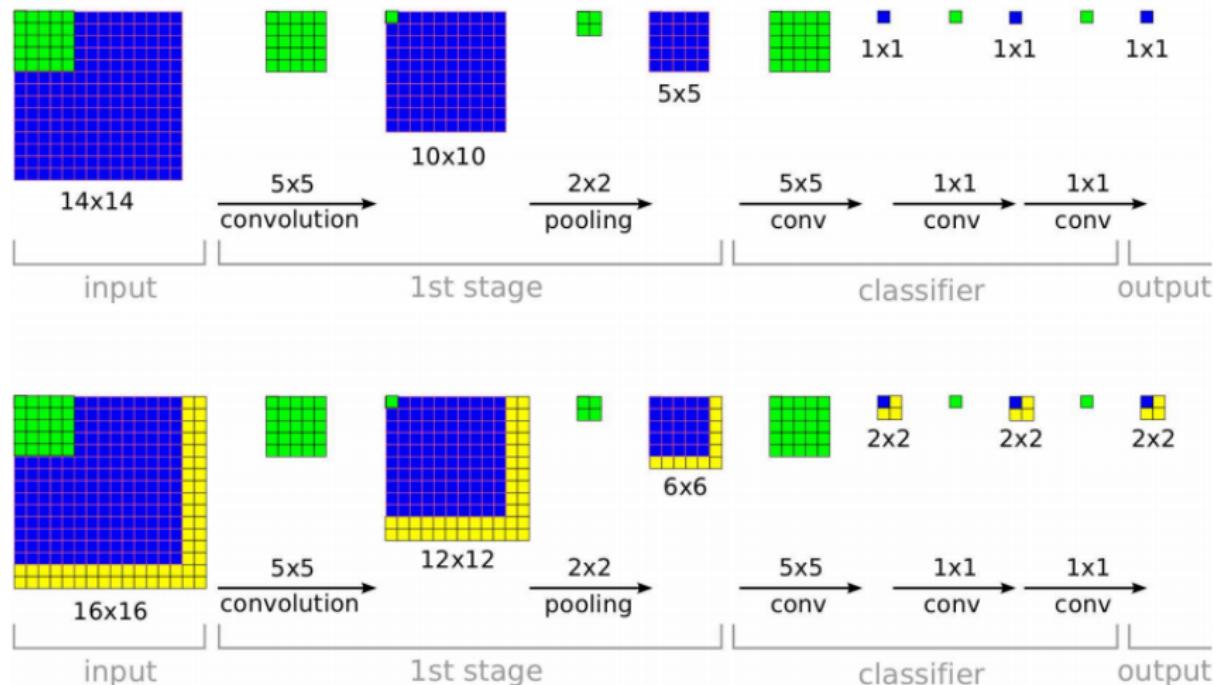
- 1 Localisation
- 2 Détection d'objet
- 3 Segmentation d'image
- 4 Estimation de pose
- 5 Reconnaissance de visage

# De la localisation à la détection d'objet

Une première idée : utiliser une fenêtre glissante multi-échelle pour localiser sur chaque sous-partie de l'image les objets présents.



# Fenêtre glissante convolutionnelle



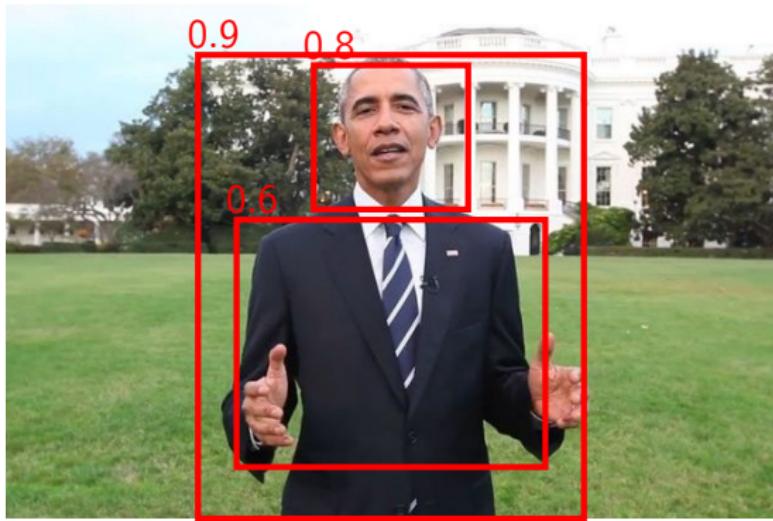
[Sermanet et al.] OverFeat : Integrated Recognition, Localization and Detection using Convolutional Networks

## Fenêtre glissante convolutionnelle



[Sermanet et al.] OverFeat : Integrated Recognition, Localization and Detection using Convolutional Networks

## Suppression des doublons



- ➊ Suppression des boîtes avec un indice de confiance  $< 0.6$ .
- ➋ Sélection parmi les boîtes restantes de la boîte  $b_{max}$  avec le plus haut indice de confiance.
- ➌ Suppression de toutes les boîtes ayant un coefficient de Jaccard  $> 0.5$  avec  $b_{max}$ .

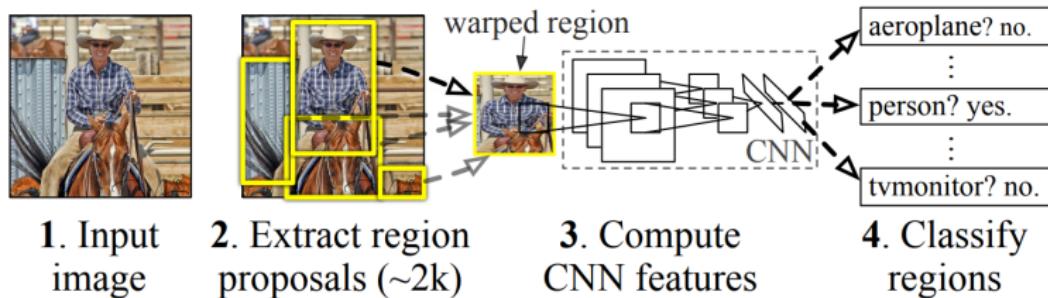
# Famille des R-CNN

Pour éviter les fenêtres glissantes : génération de régions candidates.



[Uijlings et al.] Selective Search for Object Recognition

Génération des régions candidates et classification de ces régions.

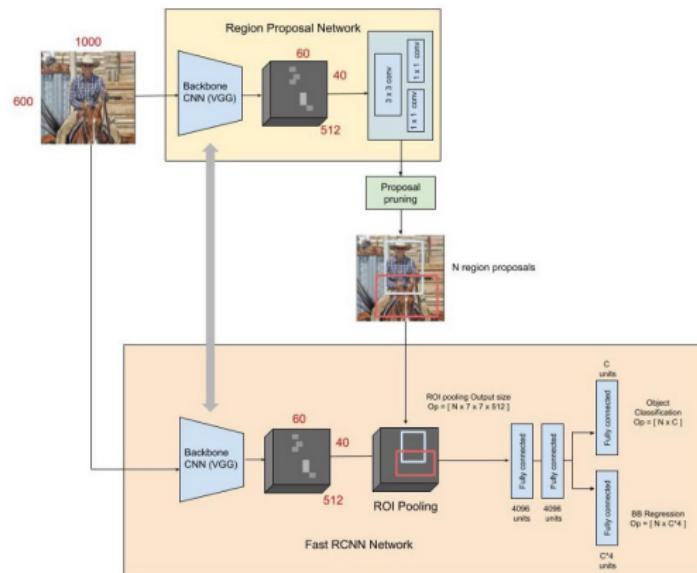


Inconvénient de ces méthodes : nécessité de faire de nombreuses prédictions par un CNN pour chaque image. **Très peu optimisé !**

[Girshick et al.] Rich feature hierarchies for accurate object detection and semantic segmentation

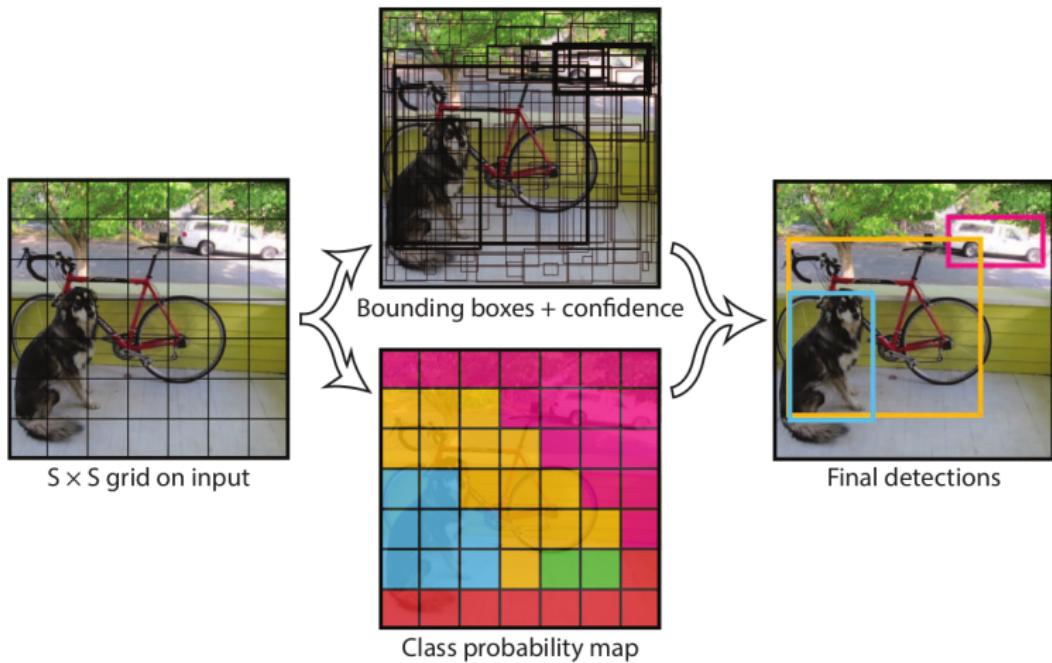
# Faster R-CNN

Un même réseau (poids partagés) pour générer des régions candidates et évaluer si elles contiennent un objet (plus rapide mais pas temps réel !)



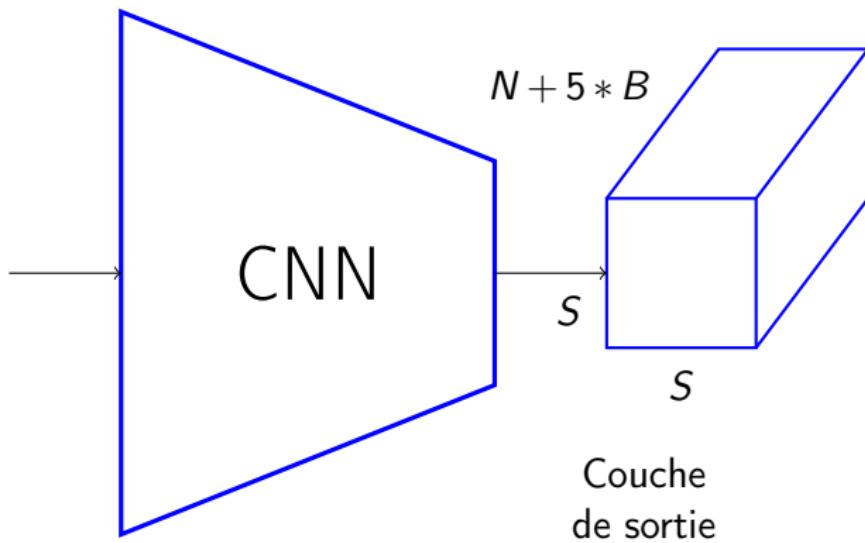
[Ren et al.] Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks

# YOLO



[Redmon et al.] You Only Look Once : Unified, Real-Time Object Detection

# YOLO



où :

- $S$  est la taille de la grille (sur PASCAL VOC,  $S = 7$ )
- $B$  est le nombre d'objets détectés par cellule (PASCAL :  $B = 2$ )
- $N$  est le nombre de classes (PASCAL :  $N = 20$ )

# YOLO : fonction de coût

$$\begin{aligned} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \quad \text{coord loss} \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left( C_i - \hat{C}_i \right)^2 \\ & \text{no box loss} \quad + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} \left( C_i - \hat{C}_i \right)^2 \\ & \text{box loss} \quad + \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \quad (3)$$

[Redmon et al.] You Only Look Once : Unified, Real-Time Object Detection

## de YOLO à YOLOv3

Un grand nombre d'améliorations sont réalisées, par exemple :

- Utilisation de "modèles" de boîtes englobantes (*anchor boxes*).
- Entraînement, et prédiction, multi-échelle.
- Évolution du CNN utilisé comme base.
- Modification de la manière d'encoder les boîtes englobantes.

et nombreuses autres subtilités...

Pour certaines, ces modifications sont reprises des autres algorithmes de détection d'objet (Faster-RCNN, SSD, etc.)

[Redmon et al.] YOLO9000 : Better, Faster, Stronger

[Redmon et al.] YOLOv3 : An Incremental Improvement

# Plan du cours

- 1 Localisation
- 2 Détection d'objet
- 3 Segmentation d'image
- 4 Estimation de pose
- 5 Reconnaissance de visage

# La base de données MS COCO



plus de 300K images, et plus de 2,5M d'instances d'objets segmentés (plus de 22 heures-humain pour 1000 segmentations)

[Lin et al.] Microsoft COCO : Common Objects in Context.

## Les défis de la segmentation



Un grand nombre de prédictions : une pour chaque pixel de l'image.

# Les défis de la segmentation



Pixels du ciel

Image originale

Pixels de la mer

La prédiction pour un pixel donné nécessite une analyse d'information locale conjointe à une compréhension plus globale de la scène.

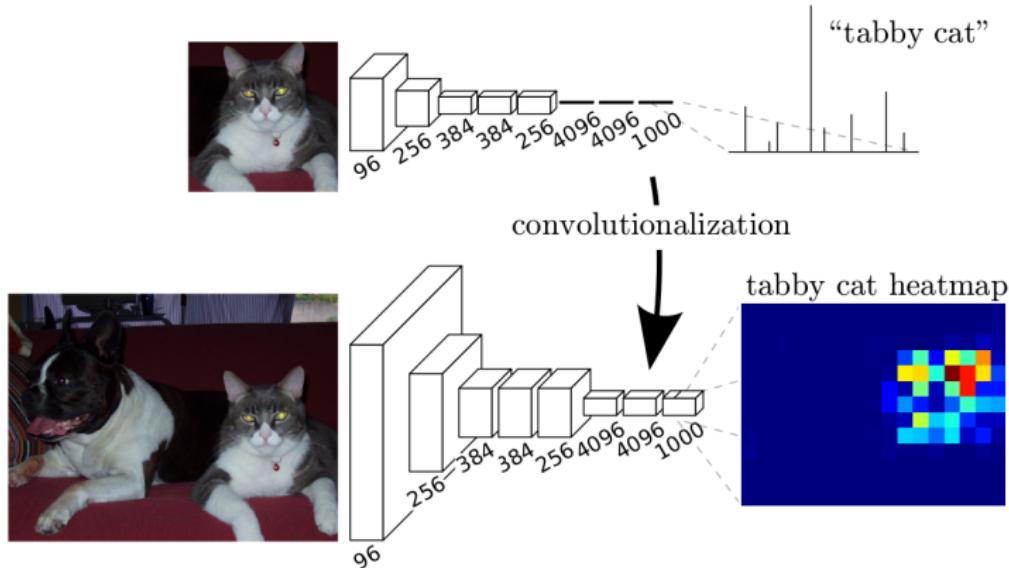
## Segmentation par fenêtre glissante



Inconvénients :

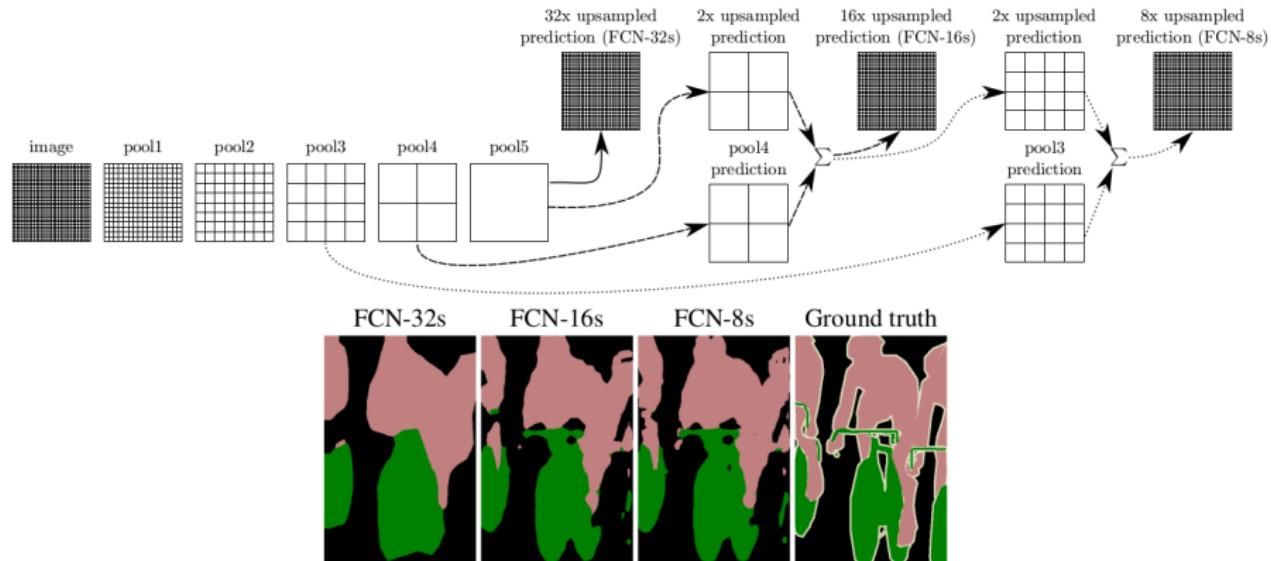
- Très coûteux.
- Pas de prise en compte de l'image globale.

# Fully Convolutional Networks (2014)



- Premier algorithme à planter une segmentation *end-to-end*.
- Repose sur une reformulation des réseaux entièrement convolutifs.
- Base VGG-16.

# Fully Convolutional Networks (2014)



- Résultat affiné en prenant en compte des prédictions intermédiaires.

[Long et al.] Fully Convolutional Networks for Semantic Segmentation.

# L'opération de déconvolution (ou convolution transposée)

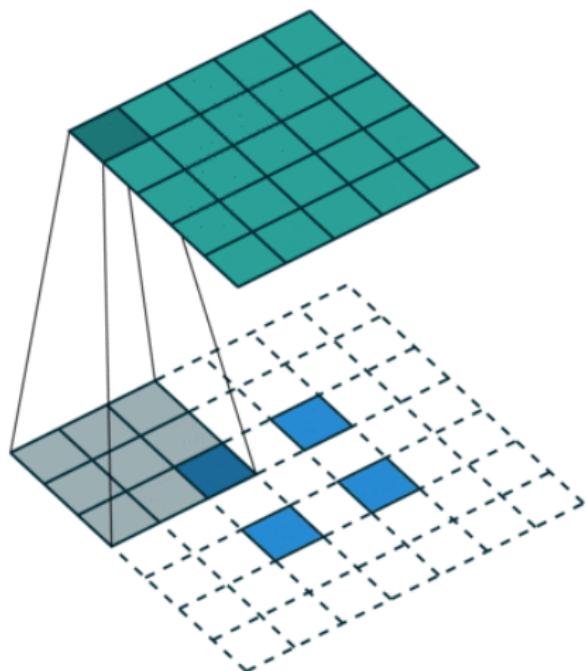
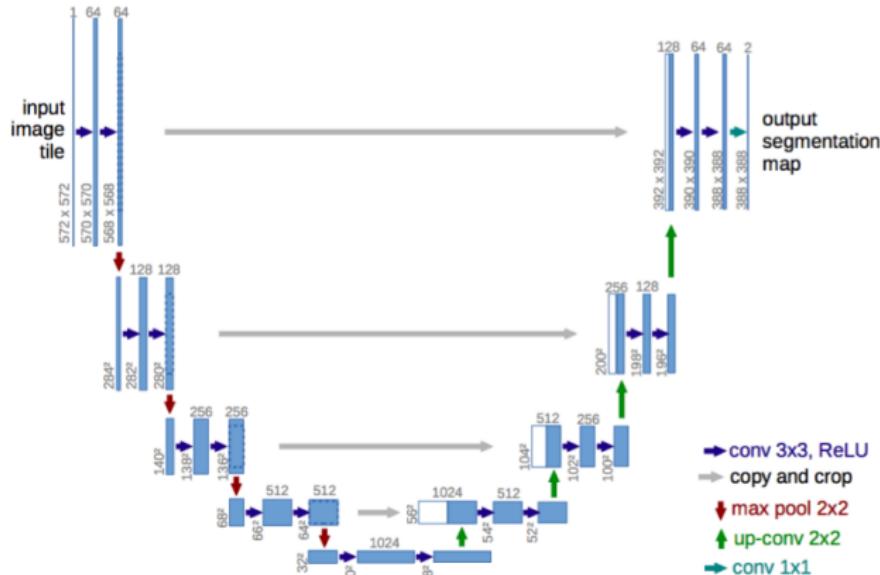


Image de [https://github.com/vdumoulin/conv\\_arithmetic](https://github.com/vdumoulin/conv_arithmetic)

## Un mot sur les fonctions de coût

- Chaque pixel porte une fonction softmax afin de déterminer quelle classe est la plus probable.
- On utilise l'entropie croisée pour comparer la classe prédite d'un pixel et la classe réelle.
- La fonction de coût finale est la moyenne des entropies croisées sur l'ensemble des pixels de l'image.

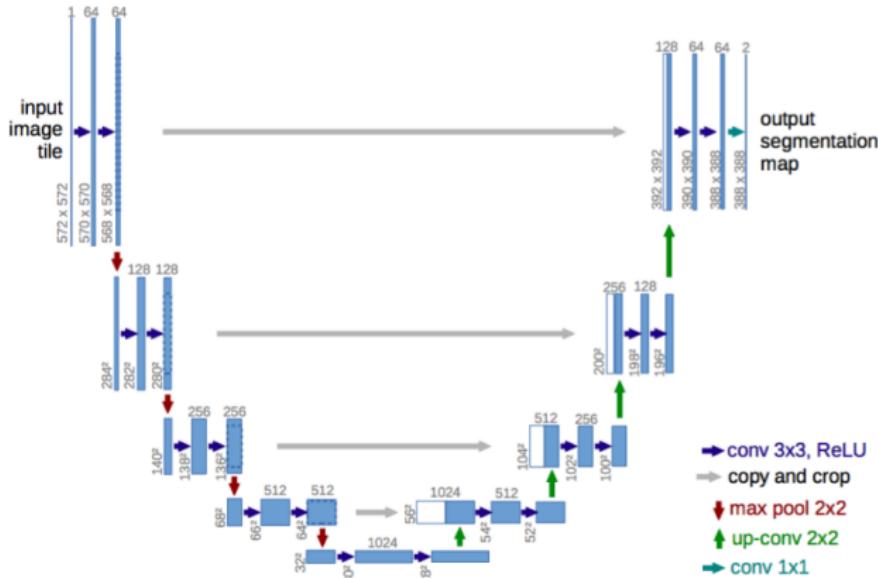
# UNet (2015)



- Une évolution de FCN.
- Probablement un des réseaux les plus utilisés pour la segmentation !

[Ronneberger et al.] U-Net : Convolutional Networks for Biomedical Image Segmentation.

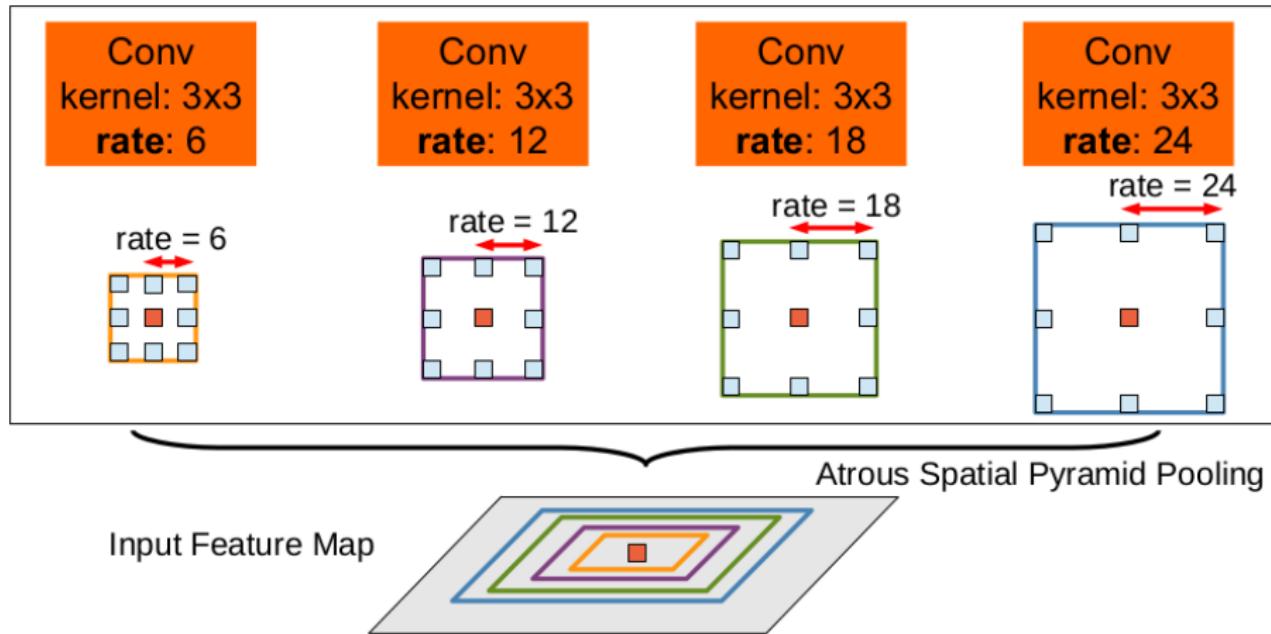
# UNet (2015)



Architecture adaptable avec n'importe quelle base convolutive !

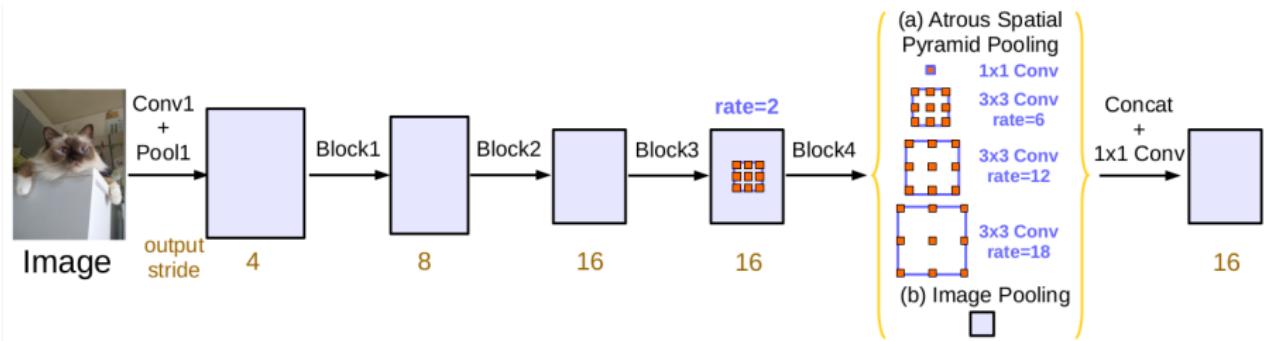
[Ronneberger et al.] U-Net : Convolutional Networks for Biomedical Image Segmentation.

# Pyramide spatiale (DeepLab v3)



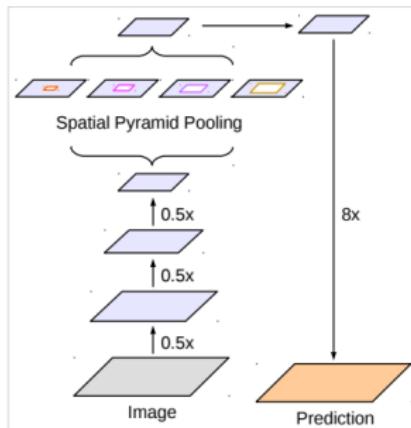
[Chen et al.] Rethinking Atrous Convolution for Semantic Image Segmentation

# DeepLab v3

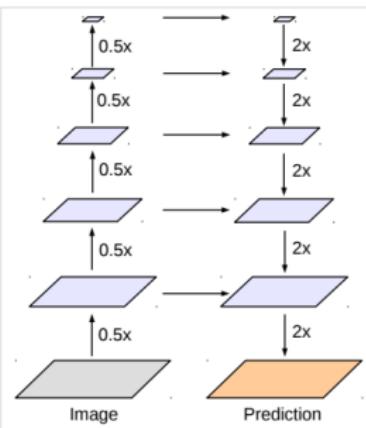


[Chen et al.] Rethinking Atrous Convolution for Semantic Image Segmentation

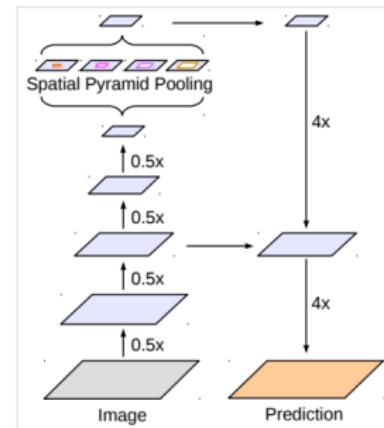
# DeepLab v3+



(a) Spatial Pyramid Pooling



(b) Encoder-Decoder



(c) Encoder-Decoder with Atrous Conv

[Chen et al.] Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation

# Bilan sur la segmentation

Deux enjeux principaux :

- **Densité de la couche de sortie** : augmentation de la dimension par deconvolution ou convolution "à trous".
- **Traitements multi-échelle** : forme en U (*auto-encoder*), pyramide spatiale.

Challenges :



# Plan du cours

1 Localisation

2 Détection d'objet

3 Segmentation d'image

4 Estimation de pose

5 Reconnaissance de visage

# Keypoints Challenge MSCOCO 2019



200000 images, 250000 personnes, 1,7M de joints.

# Estimation de pose

Problème de localisation des "joints", ou articulations, d'une personne sur une image.

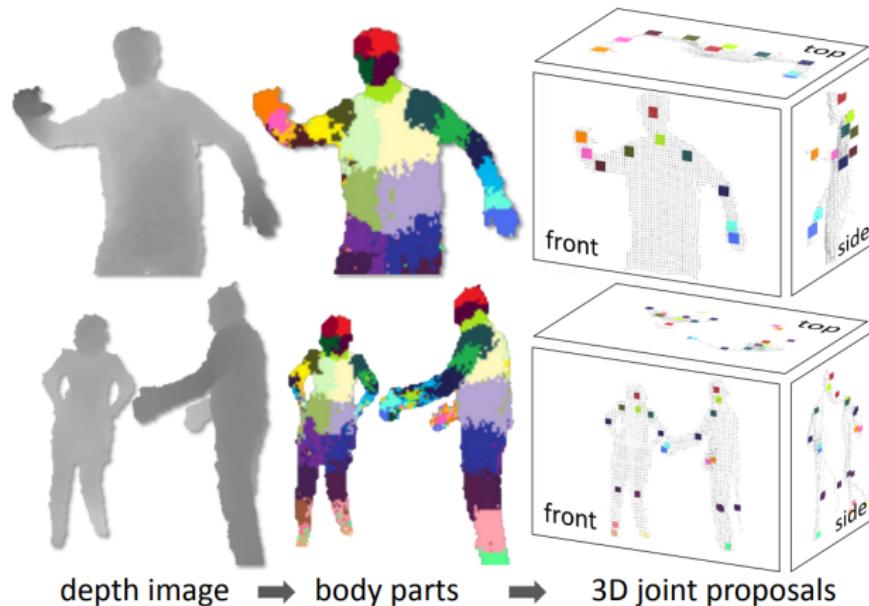


Nombreuses applications : reconnaissance d'actions, animation, jeux (Just Dance), *sports analytics*, etc.

**Difficulté** : prédictions locales mais nécessité d'une compréhension globale de la scène (comme pour la segmentation)

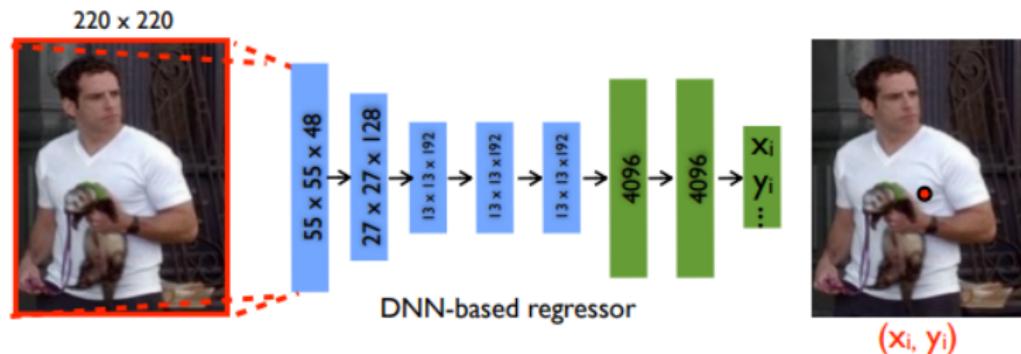
# Estimation de pose

Approche "classique" implantée dans la Kinect (2011) :



Segmentation par parties du corps (forêts aléatoires) puis estimation de mode (*mean-shift*) pour l'extraction des joints.

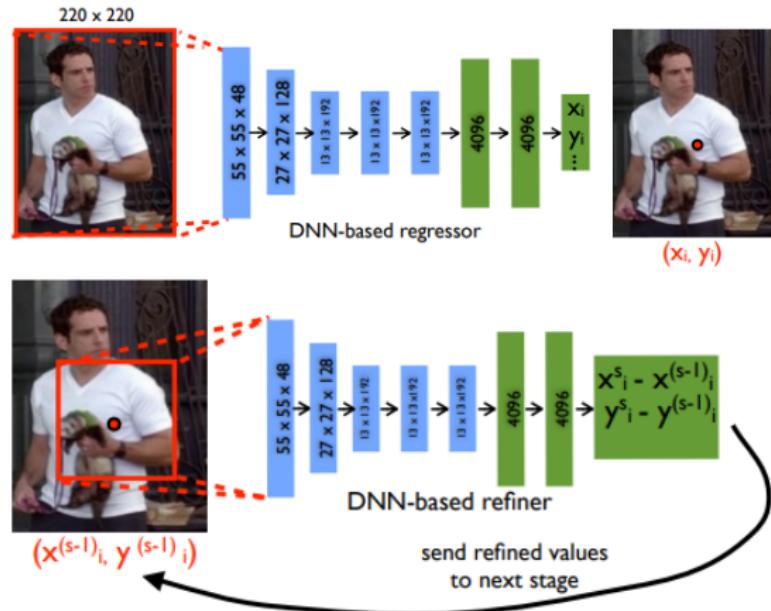
# DeepPose (2014)



Utilisation d'AlexNet comme base convolutive, prédition directe des coordonnées des joints. Augmentation **massive** des données ( $\times 40$ ) !

[Toshev et al.] DeepPose : Human Pose Estimation via Deep Neural Networks

# DeepPose (2014)

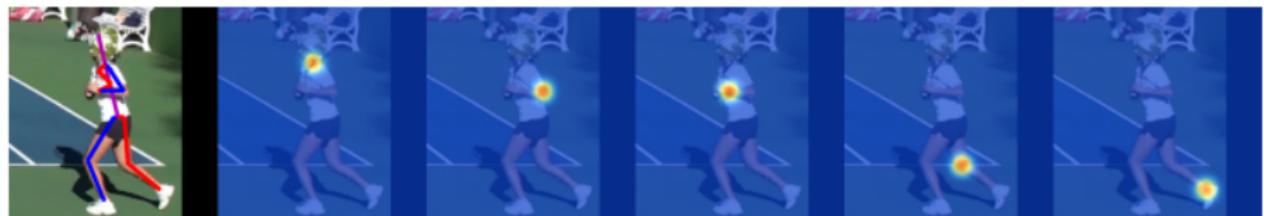


Affinage de la prédiction en repassant une sous-région de l'image dans le réseau de neurones.

[Toshev et al.] DeepPose : Human Pose Estimation via Deep Neural Networks

# Prédiction de cartes de chaleur

Une idée introduite en 2015 consiste à prédire des cartes de chaleur pour chaque joint plutôt qu'une position.



Le problème se rapproche alors d'un problème de segmentation.

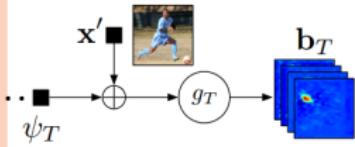
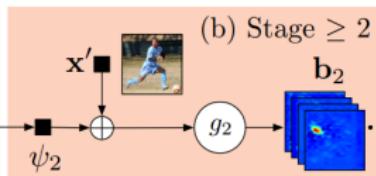
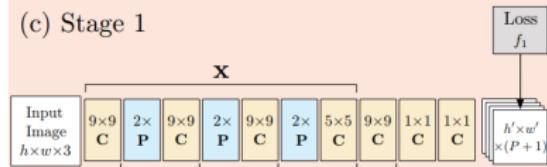
[Tompson et al.] Efficient Object Localization Using Convolutional Networks

# OpenPose (2016-2018)

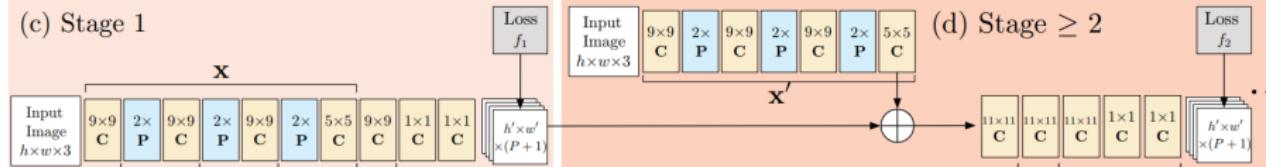
Convolutional  
Pose Machines  
( $T$ -stage)

P Pooling  
C Convolution

(c) Stage 1



(d) Stage  $\geq 2$

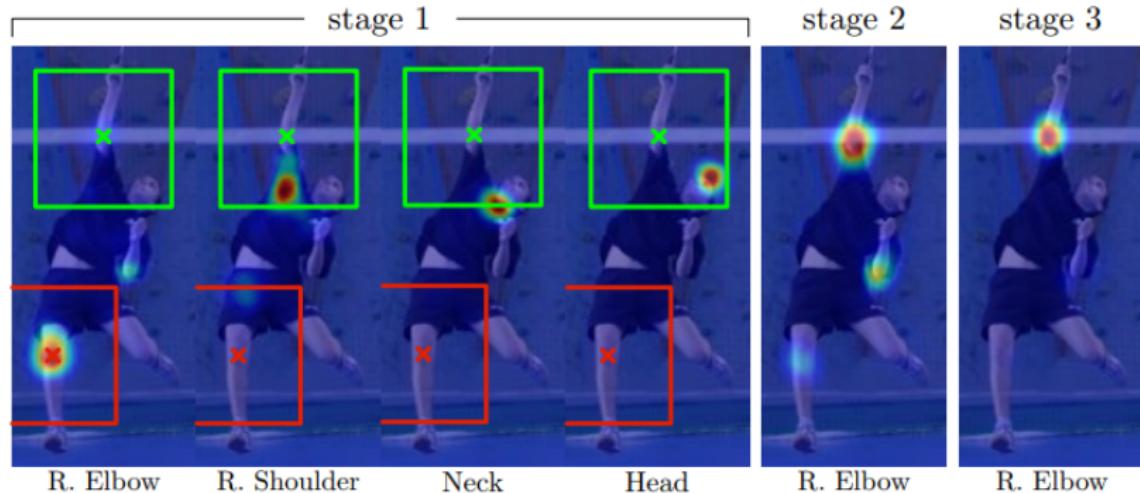


(e) Effective Receptive Field

Affinage des prédictions en prenant en compte des premières versions des prédictions.

[Wei et al.] Convolutional Pose Machines

# OpenPose (2016-2018)



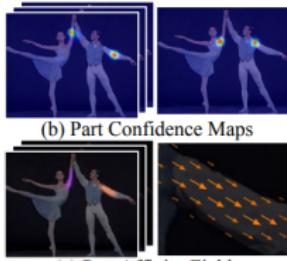
Les joints estimés avec certitude servent d'*a priori* aux joints pour lesquels il existe une ambiguïté.

[Wei et al.] Convolutional Pose Machines

# OpenPose (2016-2018)



(a) Input Image



(b) Part Confidence Maps

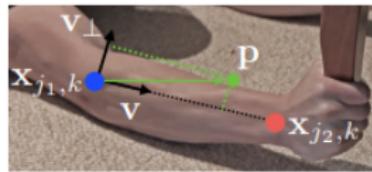
(c) Part Affinity Fields



(d) Bipartite Matching



(e) Parsing Results



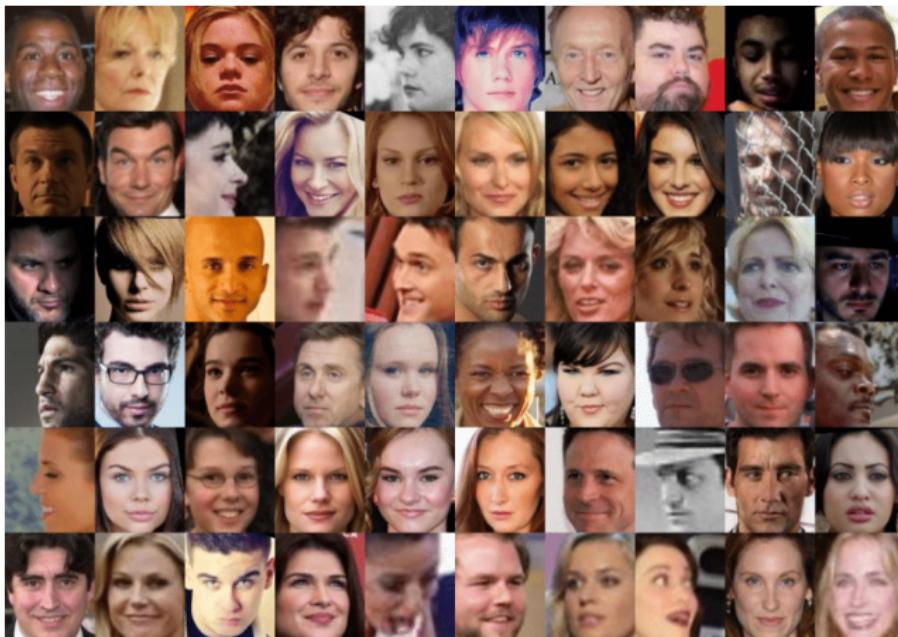
Estimation conjointe des probabilités de présence des joints, et d'un champ de vecteur permettant dans une phase de post-traitement de connecter les joints.

[Wei et al.] OpenPose : Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields

# Plan du cours

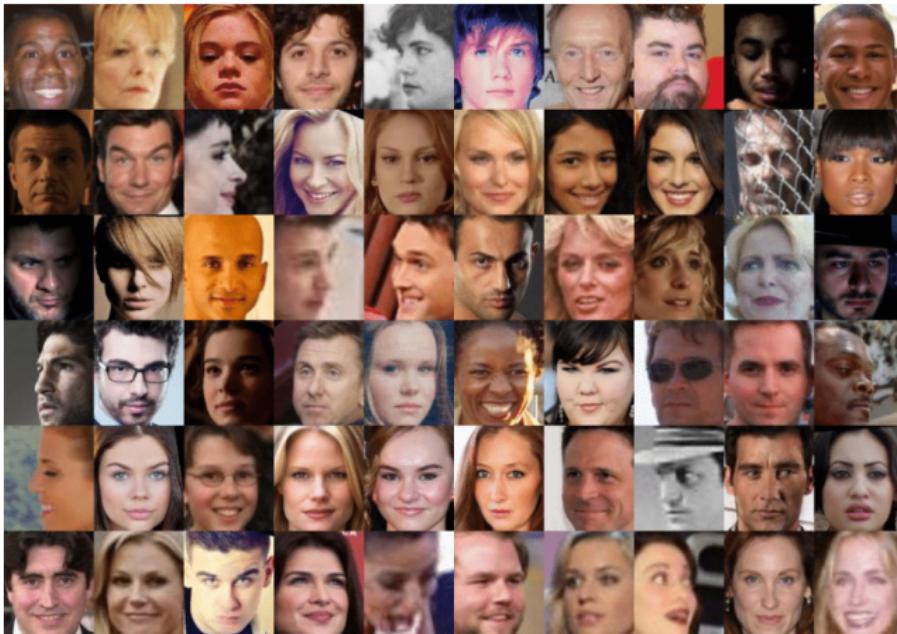
- 1 Localisation
- 2 Détection d'objet
- 3 Segmentation d'image
- 4 Estimation de pose
- 5 Reconnaissance de visage

# La reconnaissance de visage



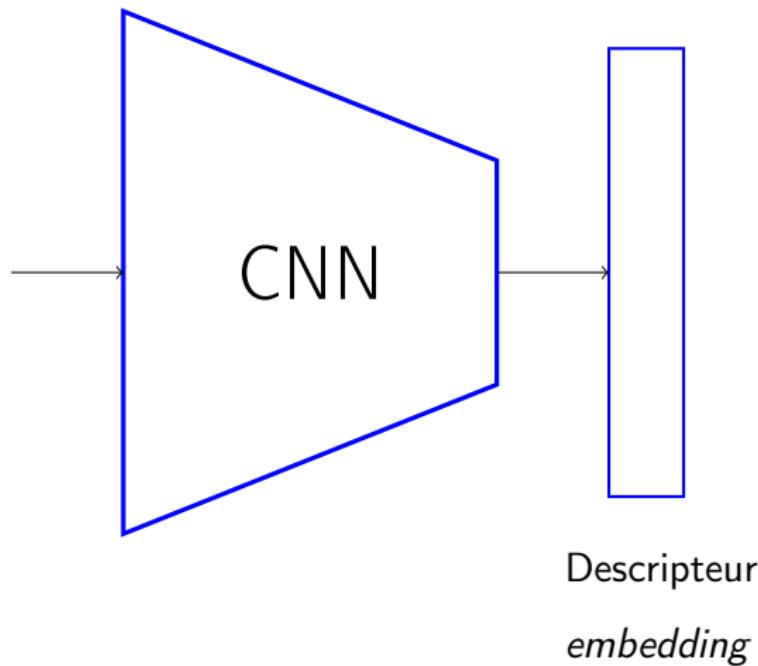
Étant donné un visage, peut-on reconnaître à qui il appartient dans une base de visages connus ?

# La reconnaissance de visage



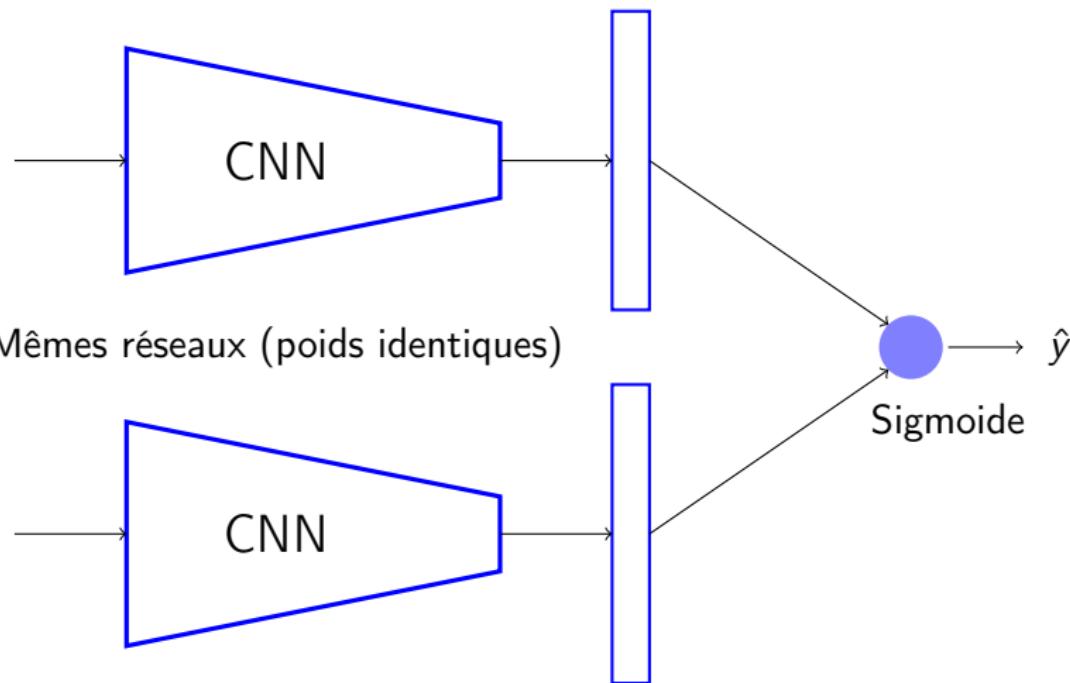
On ne peut pas formuler ce problème comme un problème de classification : il y a trop de classes, et pas assez d'exemples. Ce problème est un exemple typique de *few-shot learning*.

## Descripteur de visage (*face embedding*)



On cherche à établir un vecteur (par exemple, de taille 128) qui servira de descripteur de visage.

## Descripteur de visage (*face embedding*)



Ce réseau est entraîné avec un système dit de **réseaux siamois**. Le réseau est dupliqué et les 2 versions du réseau partagent les mêmes poids.

## DeepFace (2014)

- Un classifieur binaire est ajouté à la sortie du réseau siamois.
- Lorsque deux visages représentant la même personne sont présentés en entrée, on entraîne le système à prédire la valeur 1.
- Dans le cas contraire, si les deux visages représentant deux personnes différentes, on entraîne le système à prédire la valeur 0.

Ce système pousse le réseau à construire un espace de descripteurs dans lequel deux visages de la même personne sont plus proches que deux visages appartenant à deux visages différents.

→ Pour notre tâche de reconnaissance, il suffit ensuite de chercher le visage le plus proche (algorithme du plus proche voisin) dans l'espace des descripteurs.

[Taigman et al.] Deepface : Closing the gap to human-level performance in face verification

## FaceNet (2015)

L'algorithme FaceNet introduit une variante par rapport à DeepFace. Une fonction de coût triplet est utilisée, ce qui implique de mettre en place un réseau siamois "triple".

Pour l'entraînement, un visage ancre  $A$  est utilisé accompagné de deux autres visages : un exemple positif  $P$  représentant la même personne, et un exemple négatif  $N$  représentant une personne différente.

On utilise une fonction de coût qui vise à renforcer la propriété

$$d(A, P) + \alpha < d(A, N) \quad (1)$$

où  $\alpha > 0$  agit comme une marge.

Une contribution particulière de ce travail est le choix des triplets utilisés pour la mise à jour du réseau.

[Schroff et al.] Facenet : A unified embedding for face recognition and clustering