

الله رب العالمين

Module

Machine Learning and Computational Intelligence

MLCI

Unité d'enseignement: UEF3

Crédit: 3

Coefficient: 3

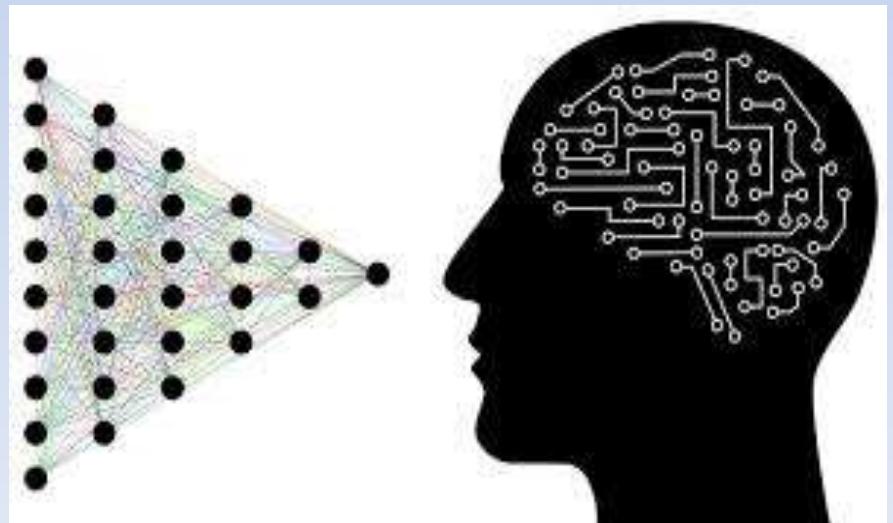
Cours: 1H30/semaine

TP: 1H30/semaine

Dr. Fergani

Baha.fergani@univ-constantine2.dz

Apprentissage profond (Deep Learning)



Architectures des réseaux de neurones

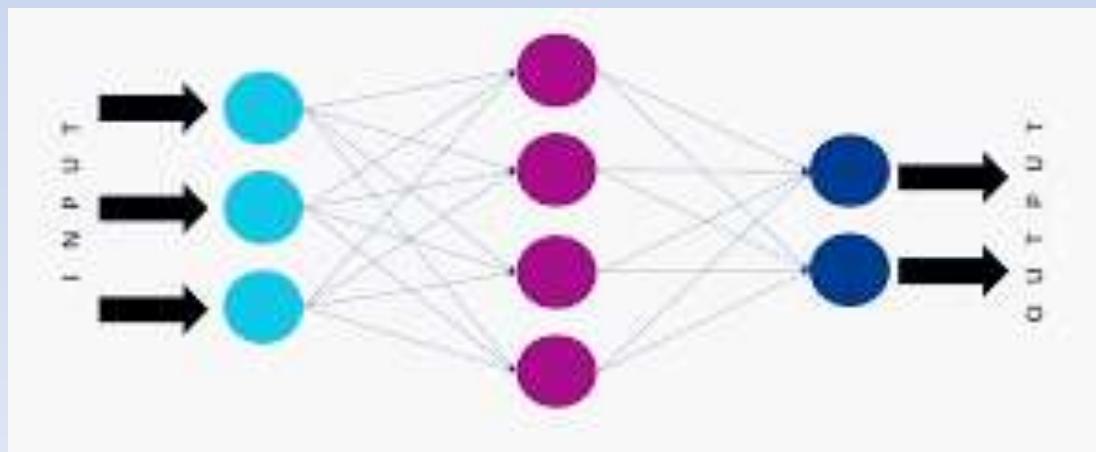
Réseau de neurones à propagation avant

Feed forward neural network

Architectures des RN

Réseau de neurones à propagation avant

- C'est un **RN** acyclique, il n'y a pas de cycles ou de boucles dans le réseau. Exemple:
Perceptron multi-couches.

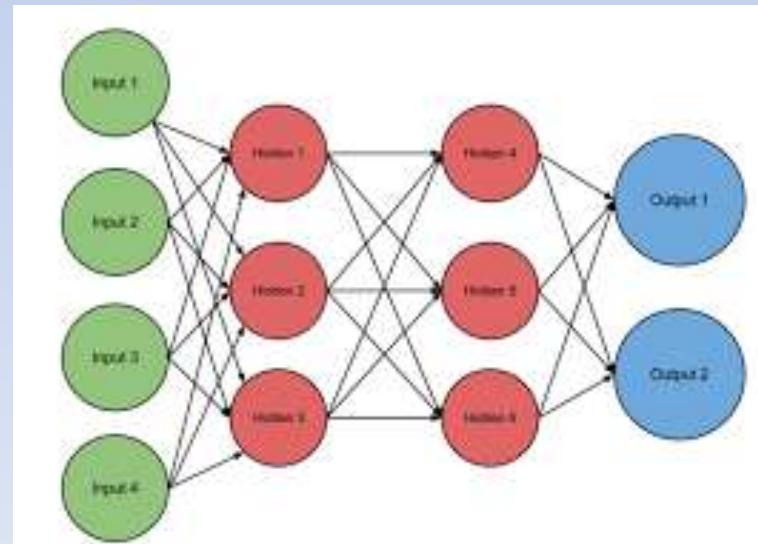
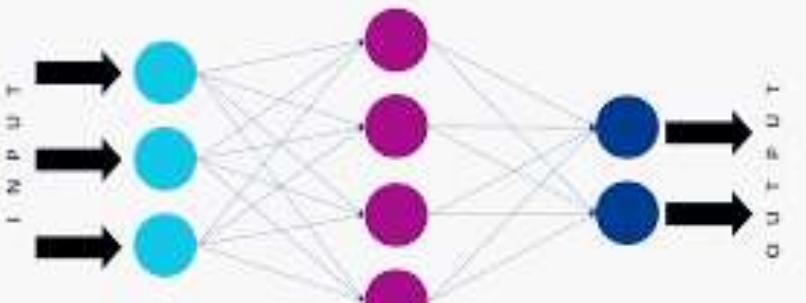


Architectures des RN

Réseau de neurones à propagation avant

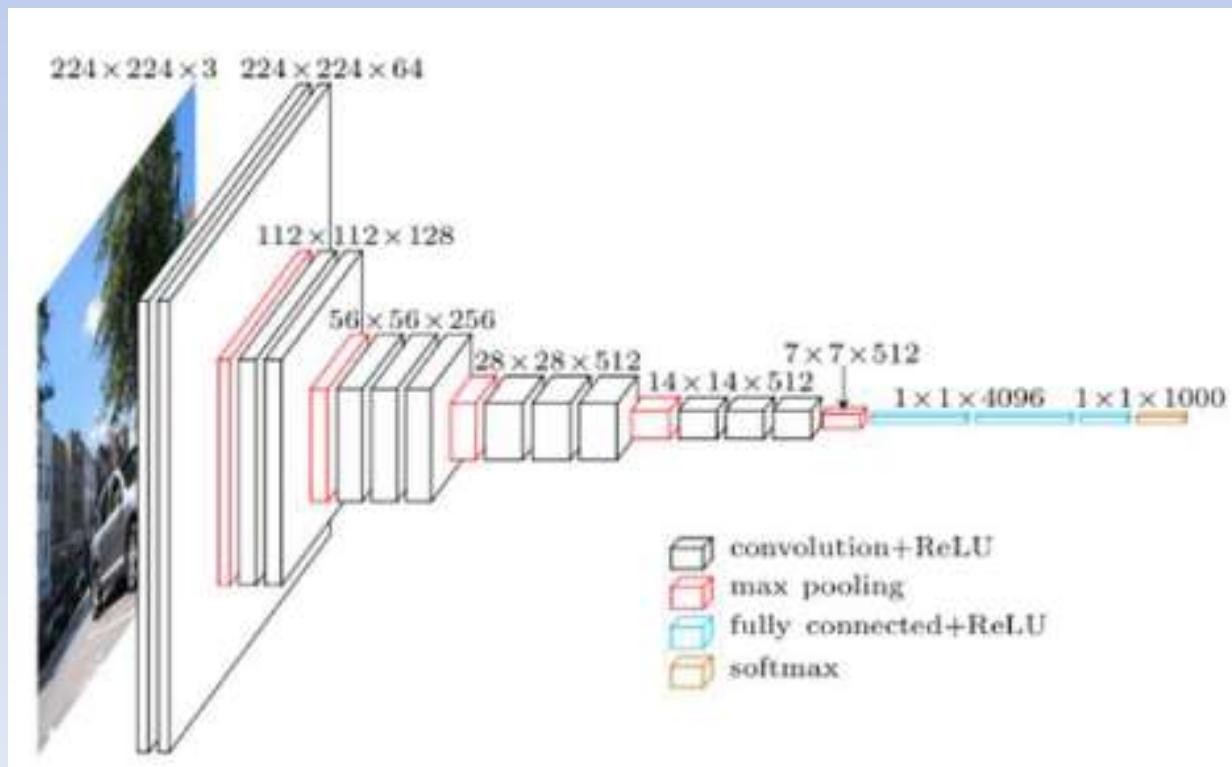
- L'information ne se déplace que dans:

Une seule direction => Vers l'avant.



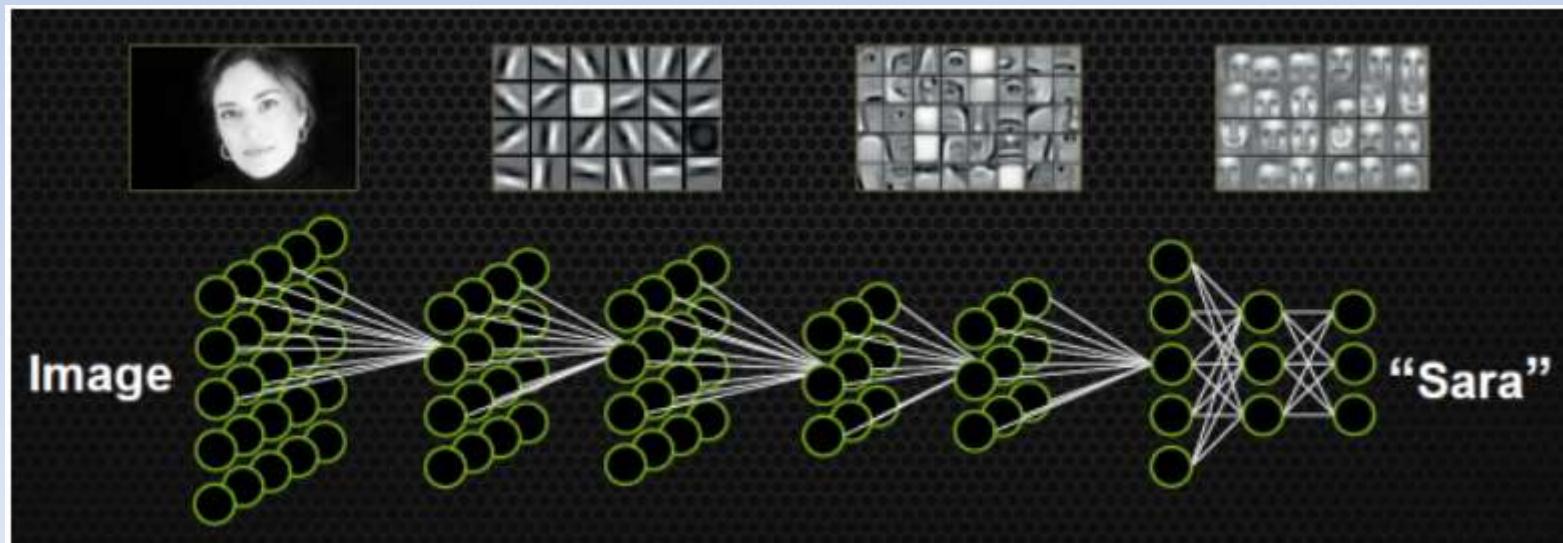
Architectures des RN

Réseau de neurones Convolutionnels

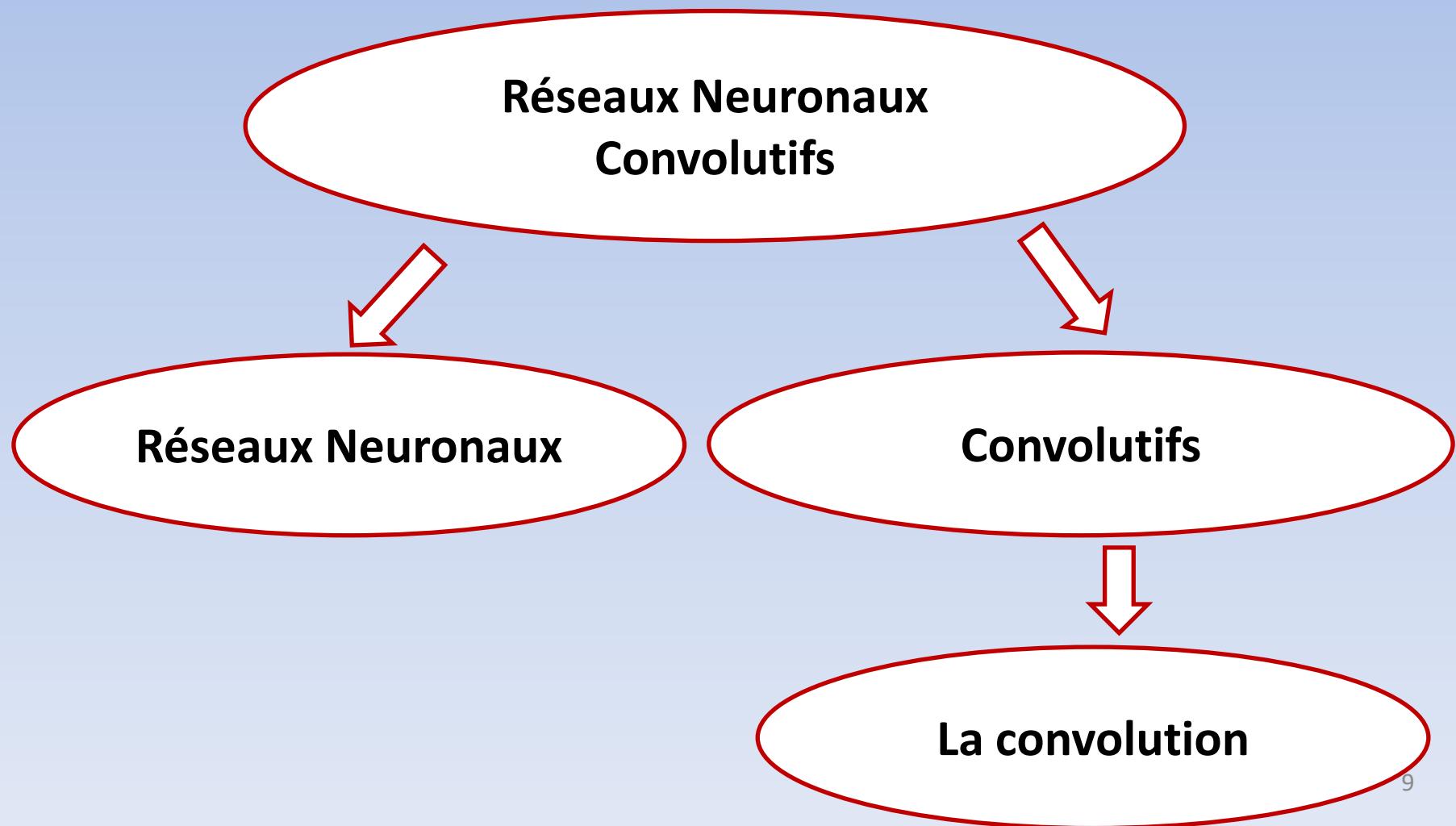


Architectures des RN

Réseau de neurones Convolutionnels



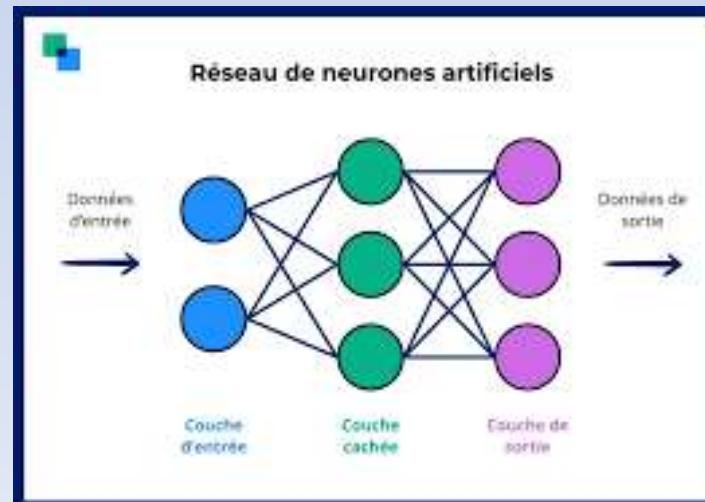
Réseau de neurones Convolutionnels



Réseau de neurones Convolutionnels

Un réseaux de neurones convolutionnel est:

- Un type de Réseau de Neurones à Propagation avant (feedforward).



Réseau de neurones Convolutionnels

Ce type de réseaux est:

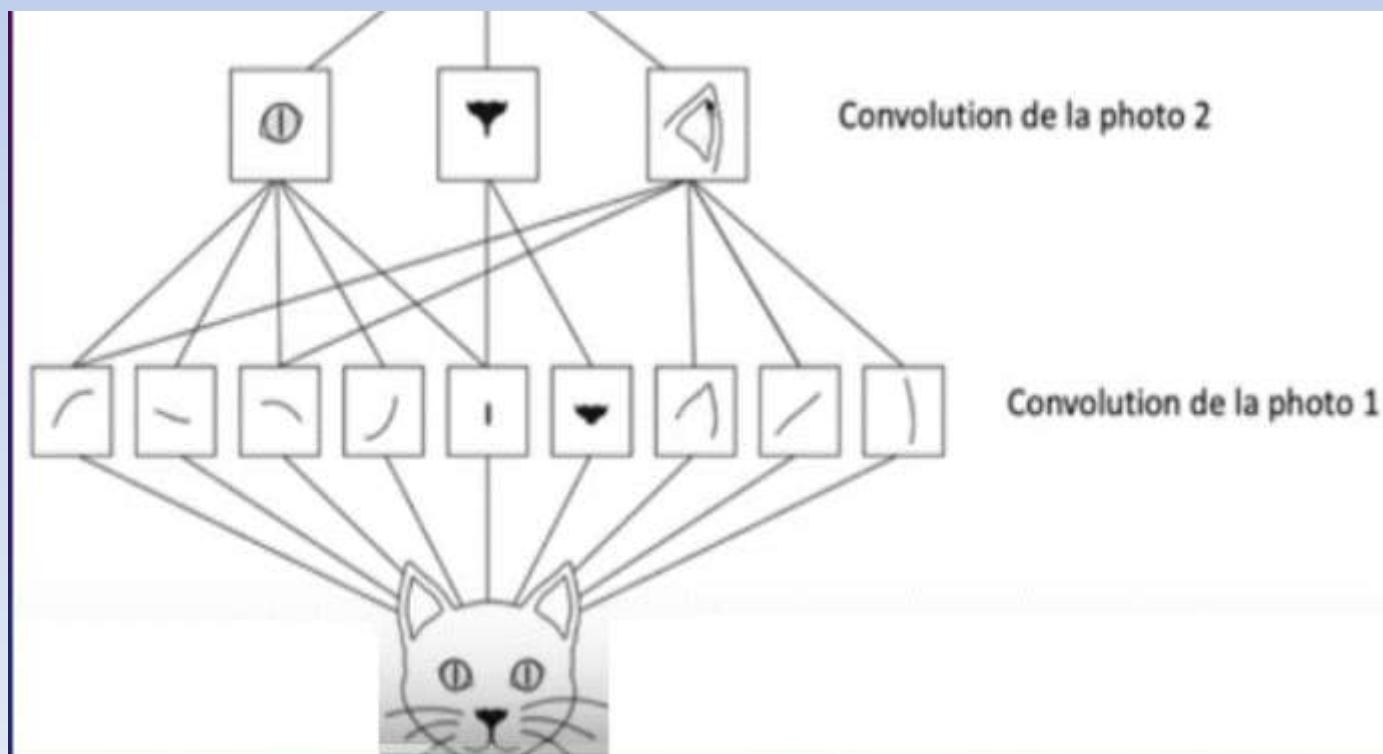
Capable d'**extraire** les caractéristiques **sans** intervention d'expert pour construire un modèle fiable d'apprentissage.



$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

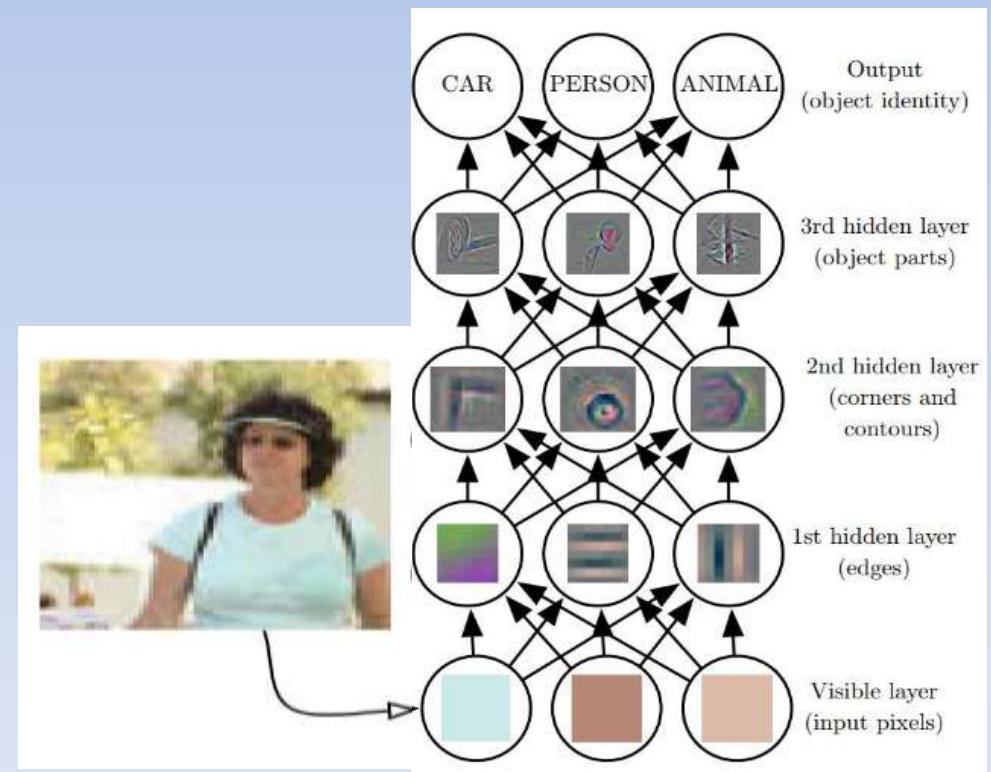
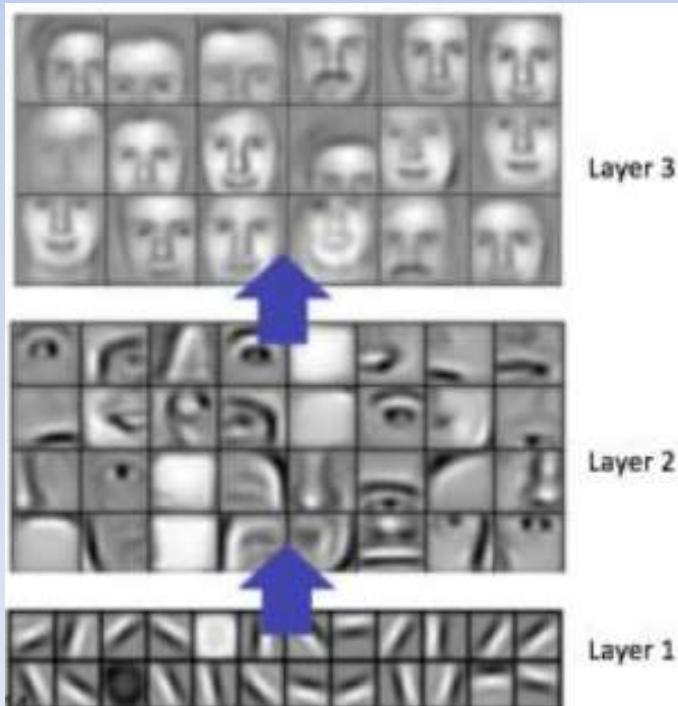
Réseau de neurones Convolutionnels

Extraction de caractéristiques.



Réseau de neurones Convolutionnels

Extraction de caractéristiques.



Réseau de neurones Convolutionnels

Remarque:

On distingue principalement 2 types de filtres:

1. Un **filtre passe-bas**: permet d'améliorer la qualité de l'image traitée.

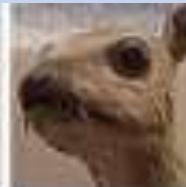


Réseau de neurones Convolutionnels

Remarque:

On distingue principalement 2 types de filtres:

1. Un **filtre passe-bas**: permet d'améliorer la qualité de l'image traitée.

$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Réseau de neurones Convolutionnels

Remarque:

On distingue principalement 2 types de filtres:

1. Un **filtre passe-bas**: permet d'améliorer la qualité de l'image traitée.



Image en entrée

Image bruitée

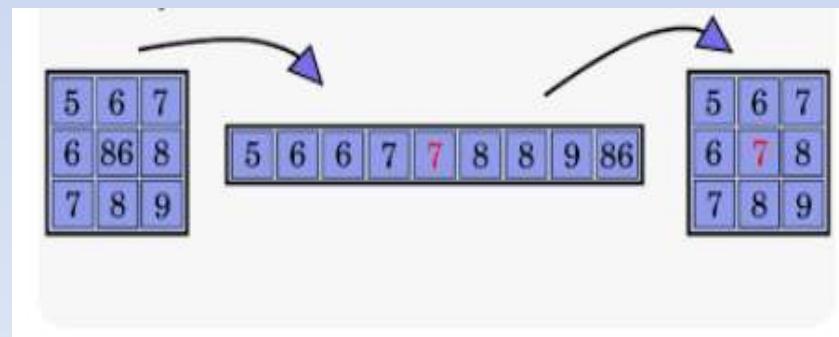
Image traitée

Réseau de neurones Convolutionnels

Remarque:

On distingue principalement 2 types de filtres:

1. Un **filtre passe-bas**: permet d'améliorer la qualité de l'image traitée.



Le filtre médian

Réseau de neurones Convolutionnels

Remarque:

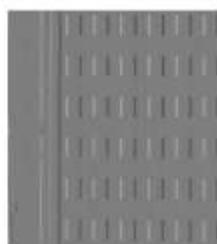
On distingue principalement 2 types de filtres:

2. **Un filtre passe-haut:** permet d'accentuer les contours de l'image traitée.

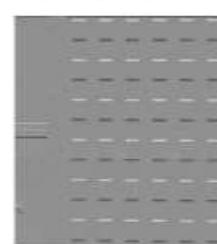
Détection bordure



Bordure
verticale

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$


Bordure
horizontale

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$


$$\begin{array}{|c|c|c|} \hline & -1 & 0 & 1 \\ \hline -1 & & & \\ \hline 0 & & & \\ \hline 1 & -1 & 0 & 1 \\ \hline \end{array}$$

F2

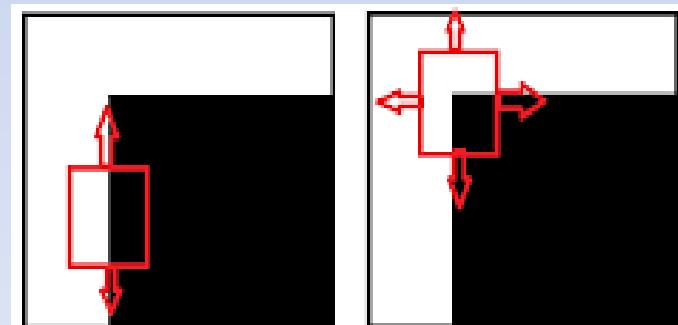
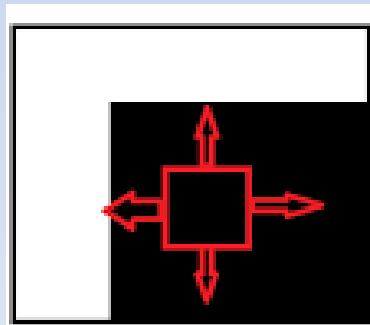


Réseau de neurones Convolutionnels

Remarque:

On distingue principalement 2 types de filtres:

2. **Un filtre passe-haut:** permet d'accentuer les contours de l'image traitée.



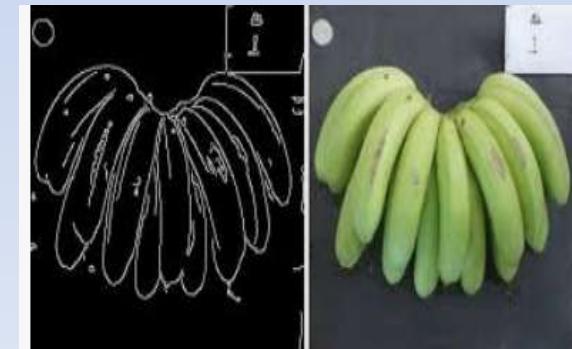
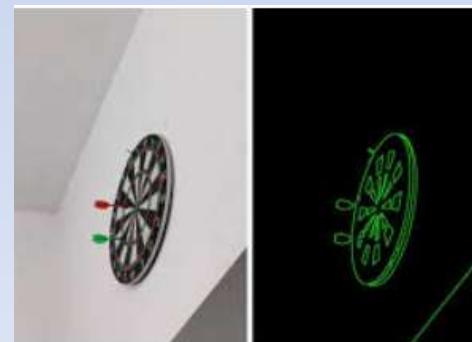
Détection de contour/ coins

Réseau de neurones Convolutionnels

Remarque:

On distingue principalement 2 types de filtres:

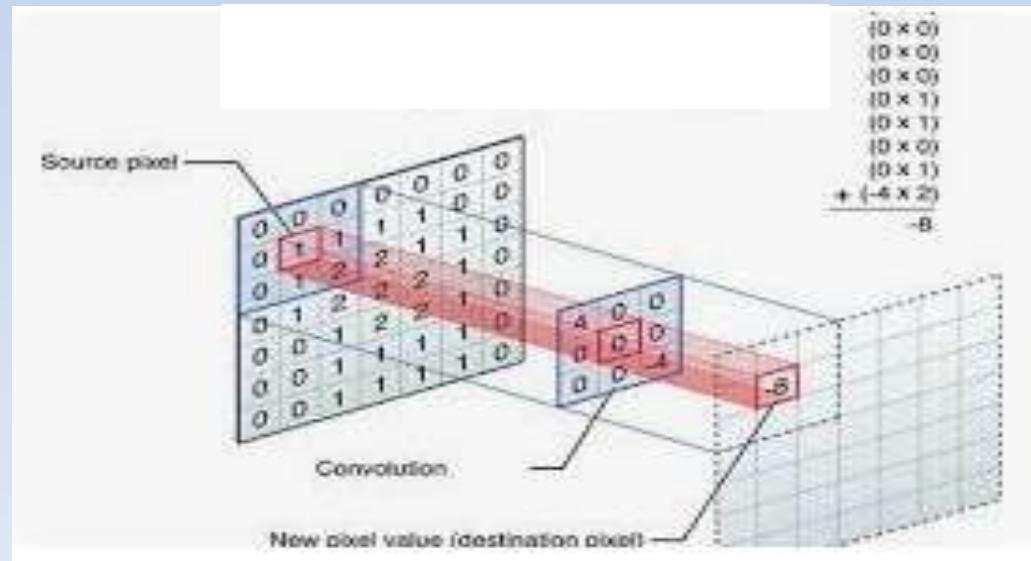
2. **Un filtre passe-haut:** permet d'accentuer les contours de l'image traitée.



Réseau de neurones Convolutionnels

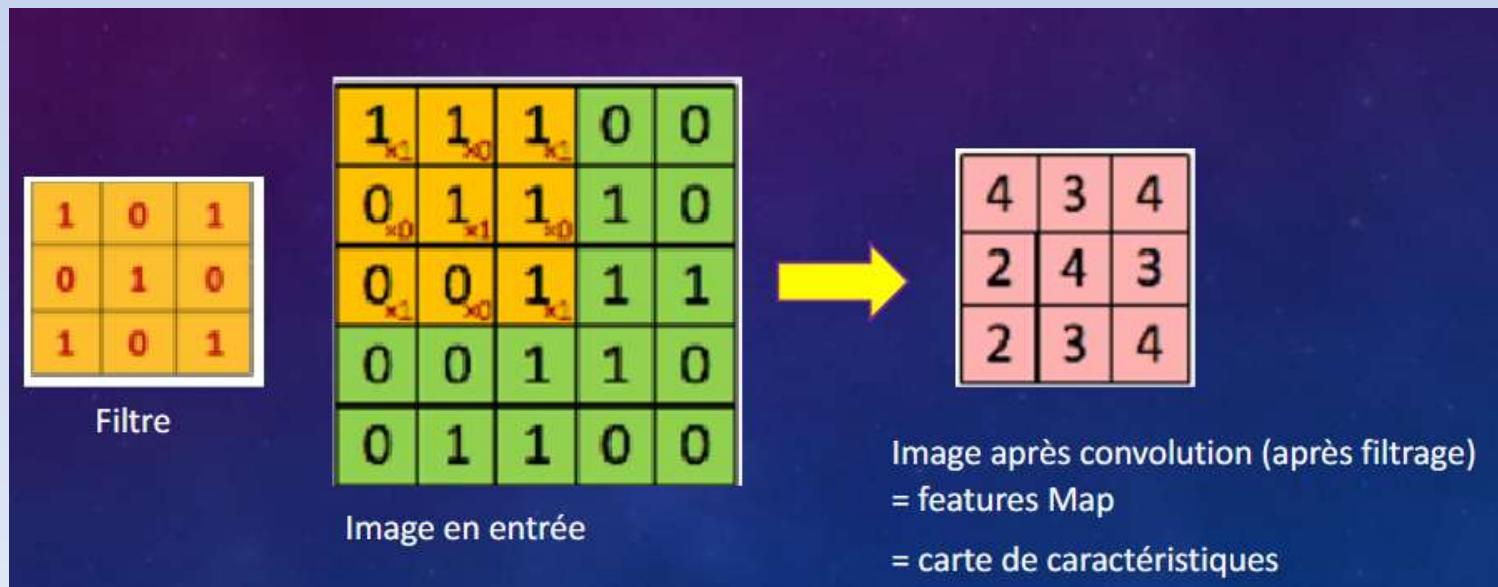
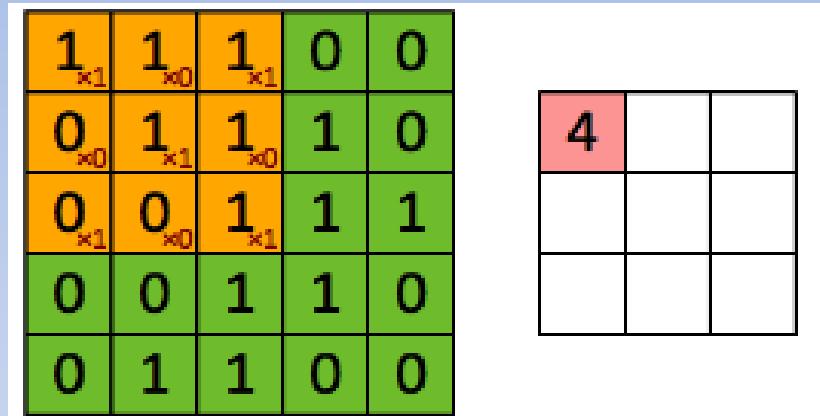
Le nom CNN vient de:

La **convolution** qui est une opération mathématique entre deux matrices que sont l'**image** et le **filtre** de convolution.



Réseau de neurones Convolutionnels

La convolution



Réseau de neurones Convolutionnels

L'opération de convolution est définie par l'équation:

$$C(i, j) = (I * N)(i, j) = \sum_{m=0}^{x-1} \sum_{n=0}^{y-1} I(i - m, j - n) \times N(m, n).$$

Réseau de neurones Convolutionnels

- Exemple:

1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

$$*[1 \ 0] =$$

7		

1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

$$*[1 \ 0] =$$

7	5	9

1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

$$*[1 \ 0] =$$

7	5	9
4	7	9
32	2	5

1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

$$*[1 \ 0] =$$

7	5	9
4	7	9
32	2	5

1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

$$*[1 \ 0] =$$

7	5	9
4	7	9
32	2	5

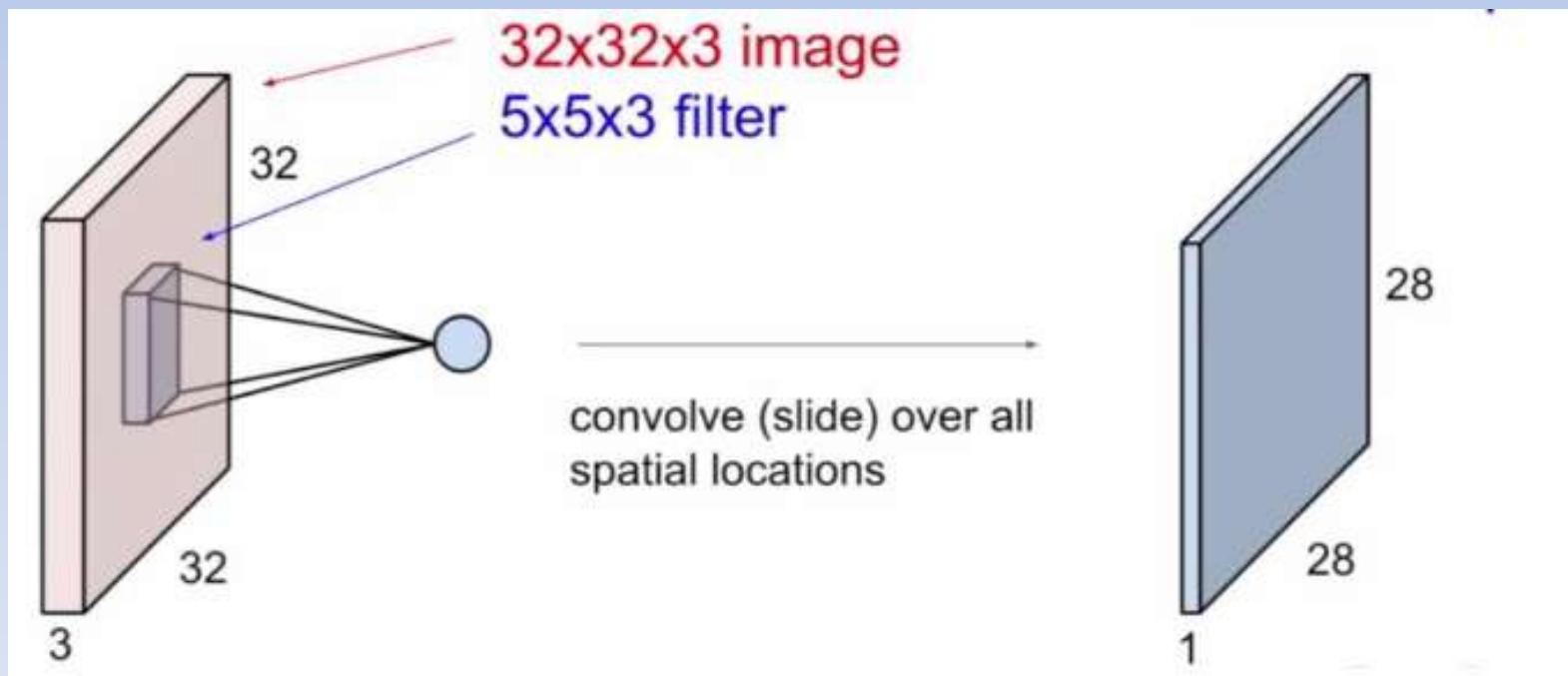
1	1	1	3
4	6	4	8
30	0	1	5
0	2	2	4

$$*[1 \ 0] =$$

7	5	9
4	7	9
32	2	5

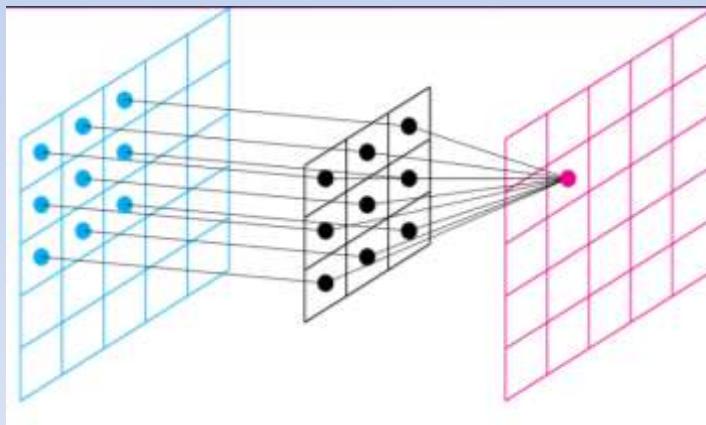
Réseau de neurones Convolutionnels

- Résultat d'appliquer un filtre:

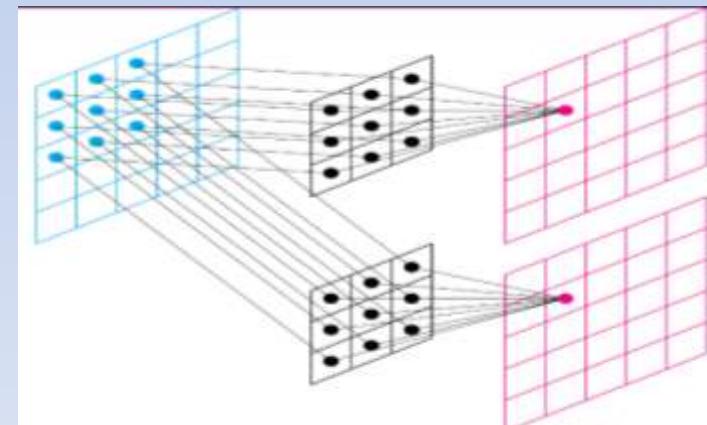


Réseau de neurones Convolutionnels

La convolution répétée plusieurs fois permet d'extraire les caractéristiques les plus pertinentes de l'image dans un vecteur.



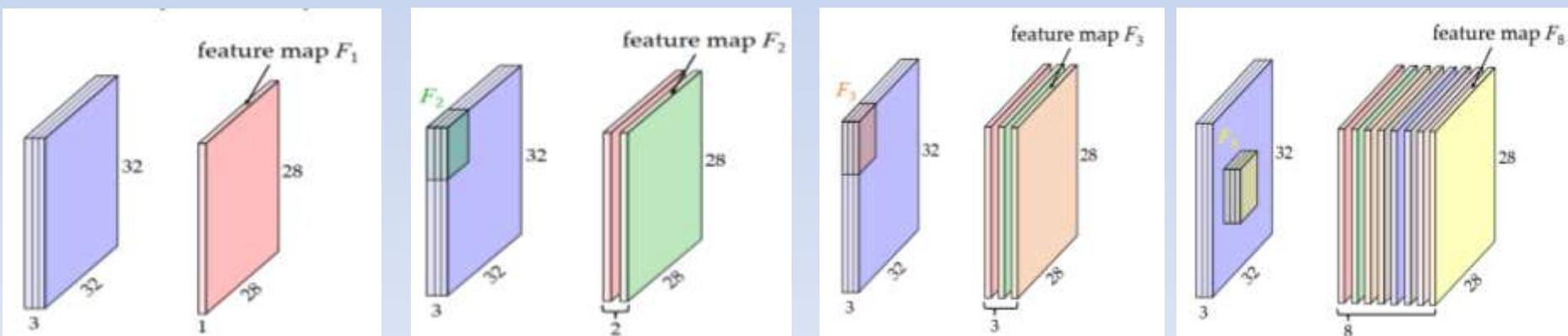
Un seul masque de convolution est appliqué



Deux masques de convolution sont appliqués

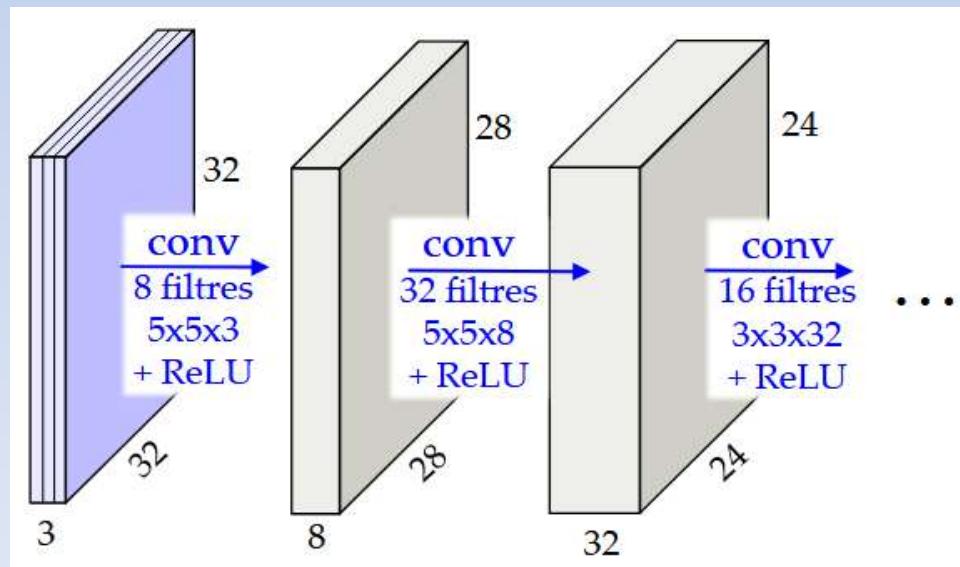
Réseau de neurones Convolutionnels

La convolution répétée plusieurs fois permet d'extraire les caractéristiques les plus pertinentes de l'image dans un vecteur.

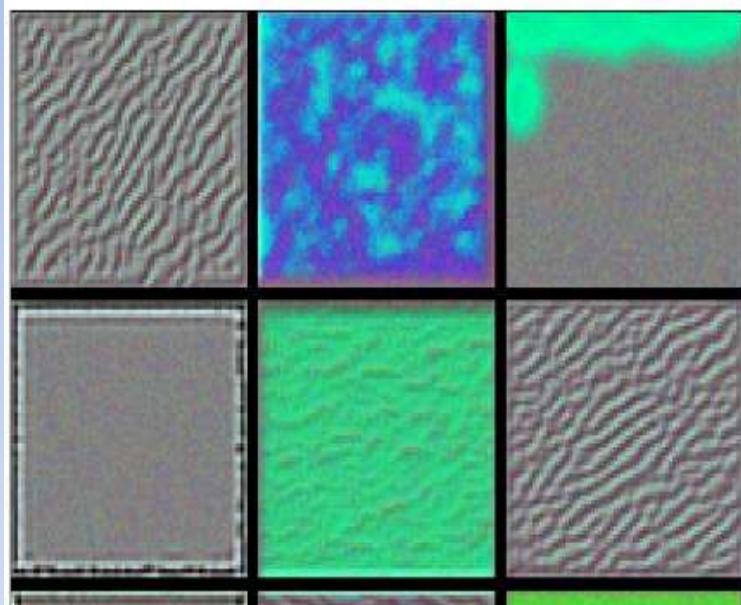


Réseau de neurones Convolutionnels

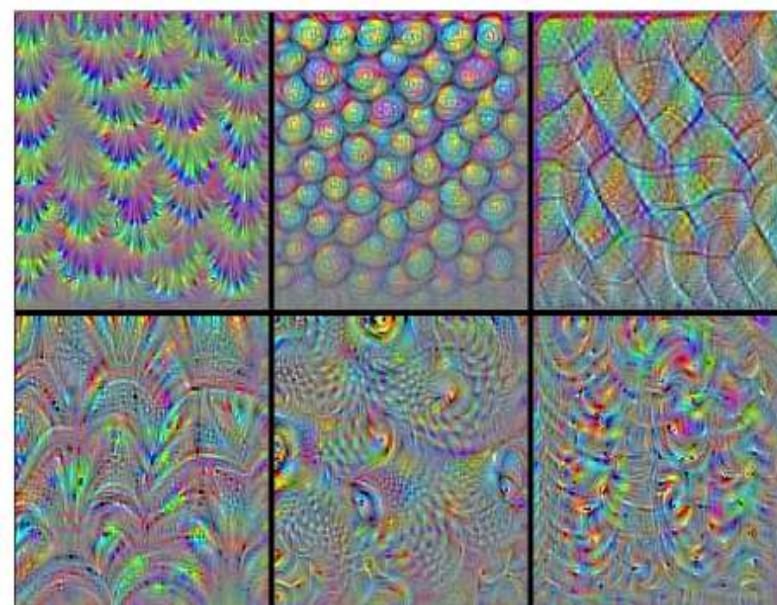
La convolution répétée plusieurs fois permet d'extraire les caractéristiques les plus pertinentes de l'image dans un vecteur.



Réseau de neurones Convolutionnels



Première couche : concepts simples,
grossiers

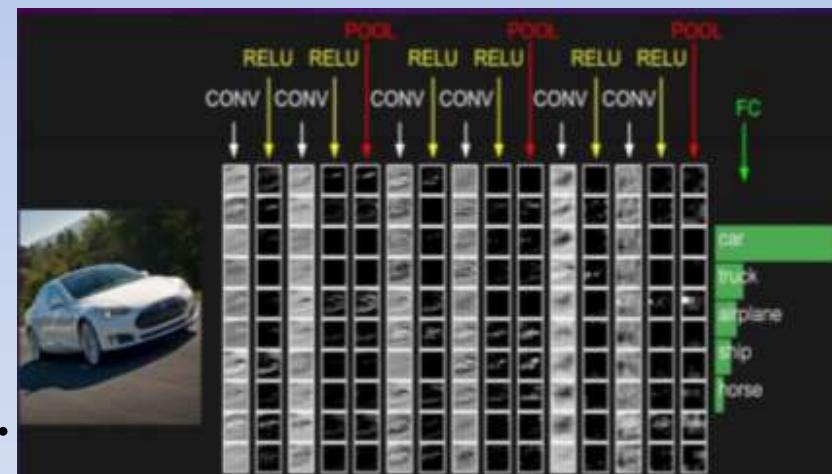


Dernière couche : concepts plus abstraits
(textures, formes, ...)

Réseau de neurones Convolutionnels

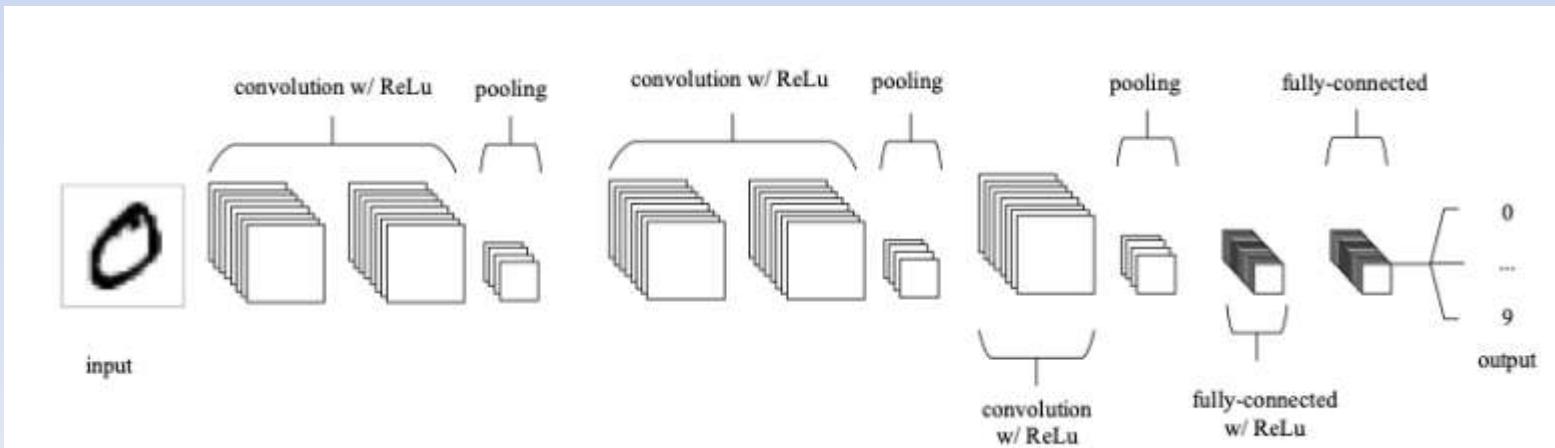
L'architecture de base du CNN comporte les couches :

1. La couche de convolution.
2. La couche correction ReLU
3. La couche de regroupement.
4. La couche d'aplatissement (flattening).
5. La couche entièrement connectée.



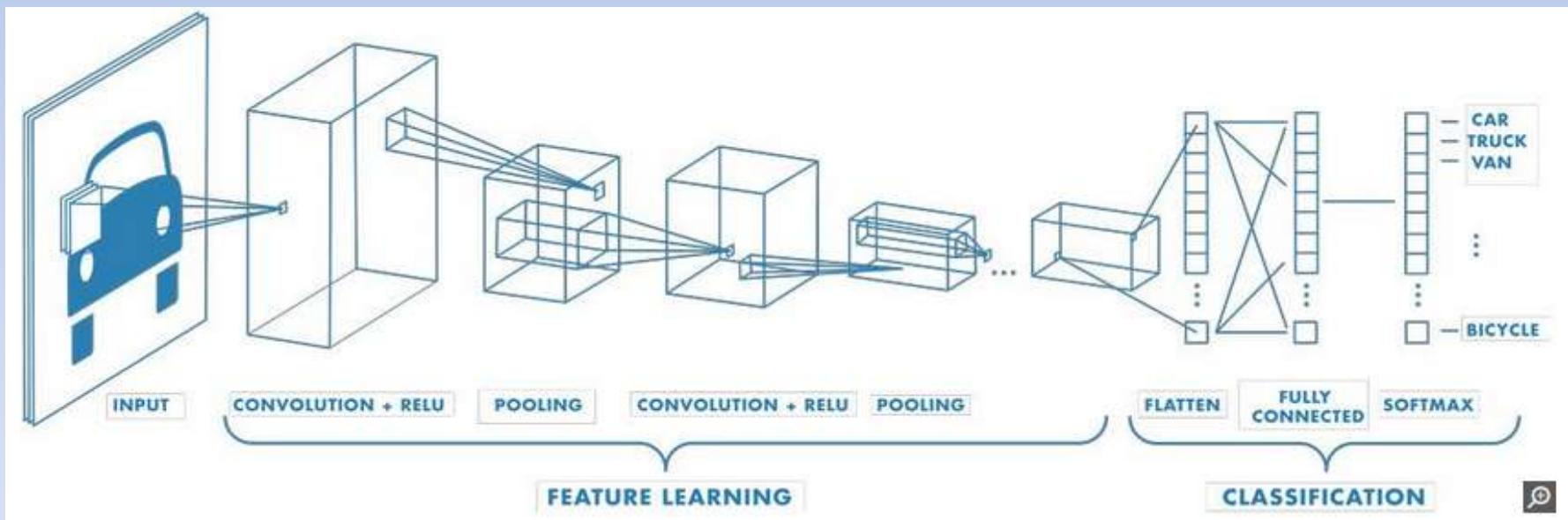
Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte plusieurs couches :



Réseau de neurones Convolutionnels

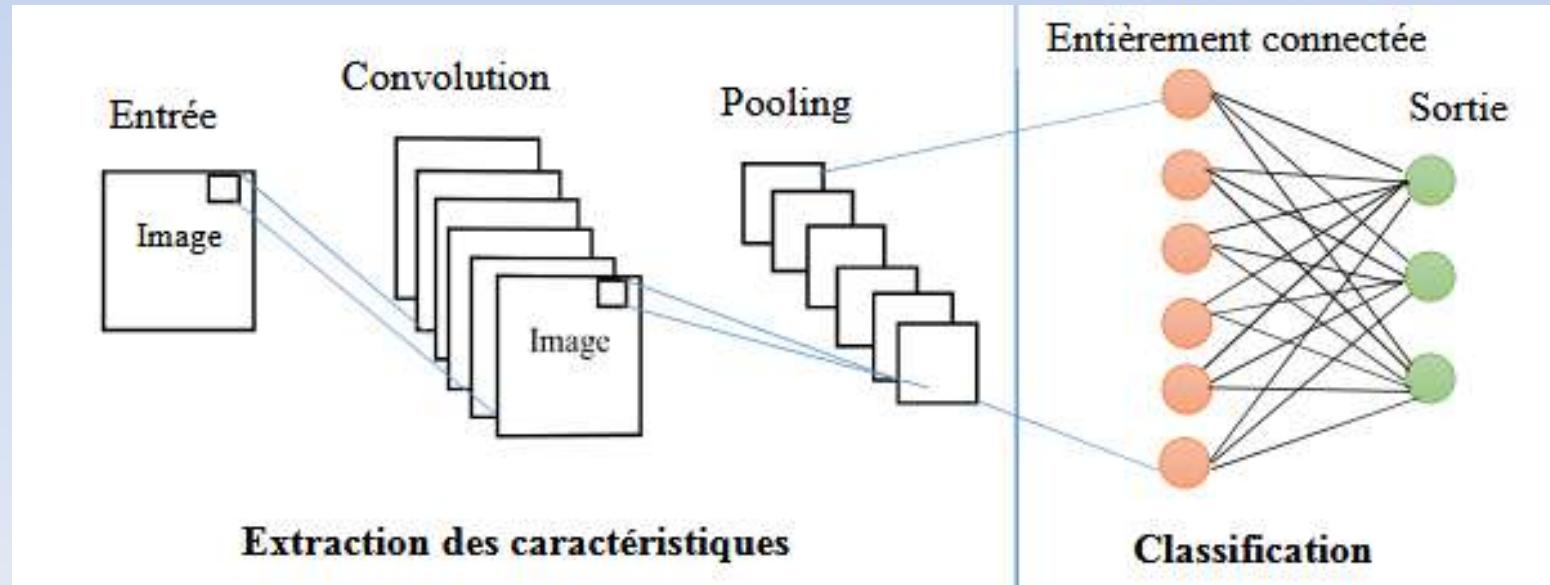
L'architecture de base du CNN comporte plusieurs couches :



Réseau de neurones Convolutionnels

1. La couche de convolution:

Elle permet l'extraction des caractéristiques plus pertinentes.



Réseau de neurones Convolutionnels

1. La couche de convolution:

En utilisant un nombre de filtres:

- Filtre sobel.



Sortie: une carte de caractéristique (feature Map).

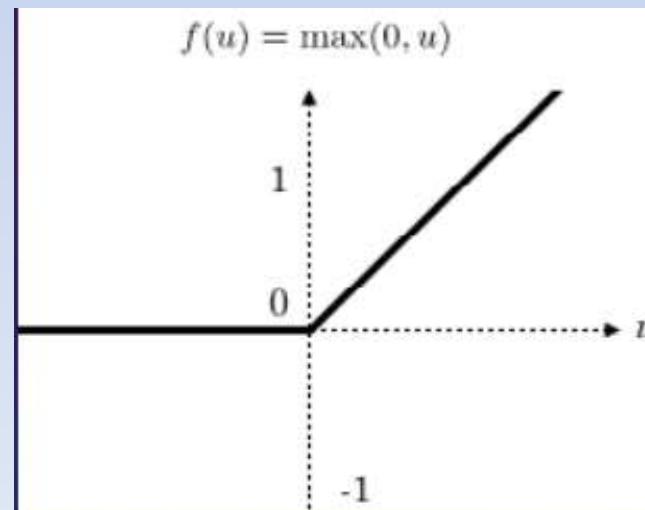
Réseau de neurones Convolutionnels

- Les **noyaux** des filtres désignent les **poids** de la couche de convolution.
- Ils sont initialisés puis mis à jour par la rétro-propagation.

Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

2. **La couche correction ReLU:** une fonction qu'on doit l'appliquer à chaque pixel d'une image après convolution.

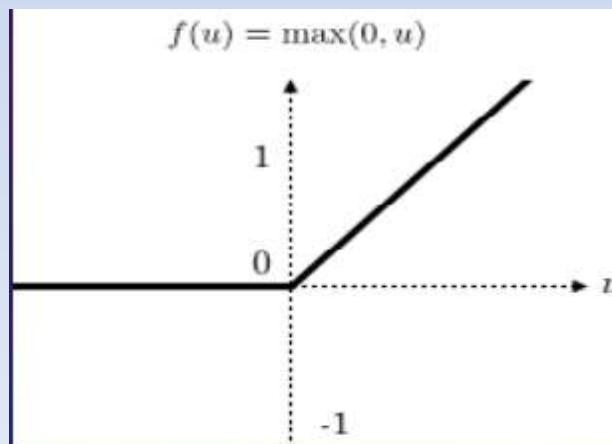


Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches:

2. La couche correction ReLU: Elle remplace chaque valeur négative par un 0.

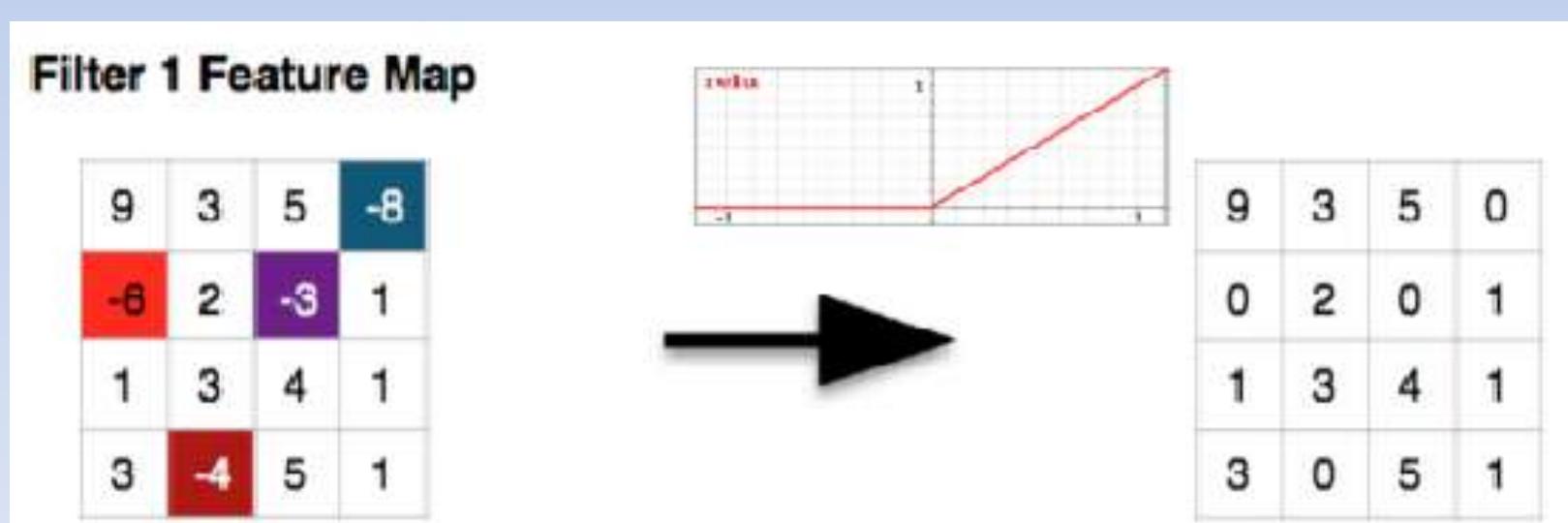
- Si elle n'est pas appliquée, la fonction créée sera linéaire.



Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches

2. La couche correction ReLU:

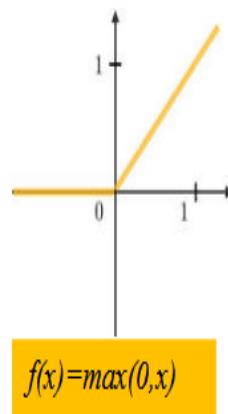


Réseau de neurones Convolutionnels

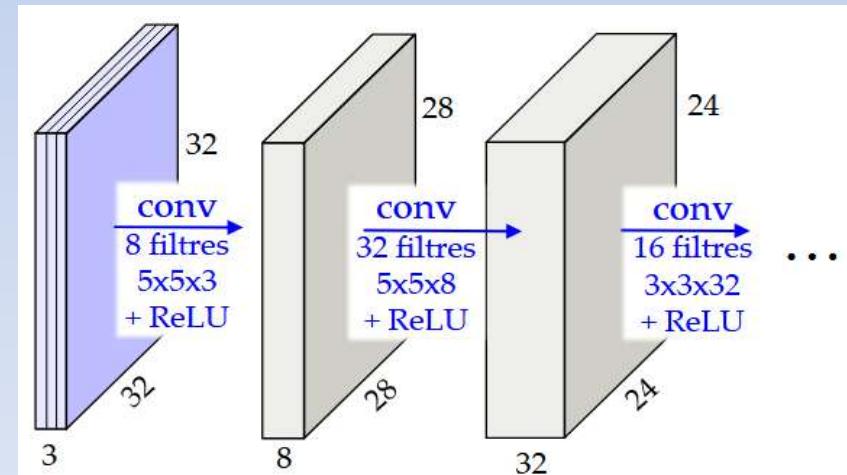
L'architecture de base du CNN comporte les couches

2. La couche correction ReLU:

23	6	-8	-1	5
-9	-5	13	4	7
22	4	17	8	8
-4	-8	6	-3	6
1	6	-4	-5	1



23	6	0	0	5
0	0	13	4	7
22	4	17	8	8
0	0	6	0	6
1	6	0	0	1



Réseau de neurones Convolutionnels

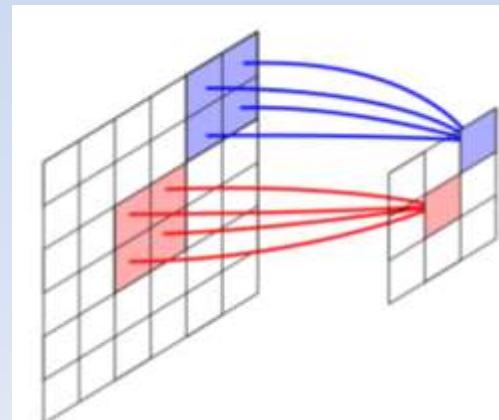
Remarque:

Aucune fonction d'activation n'est appliquée dans la couche de convolution.

Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

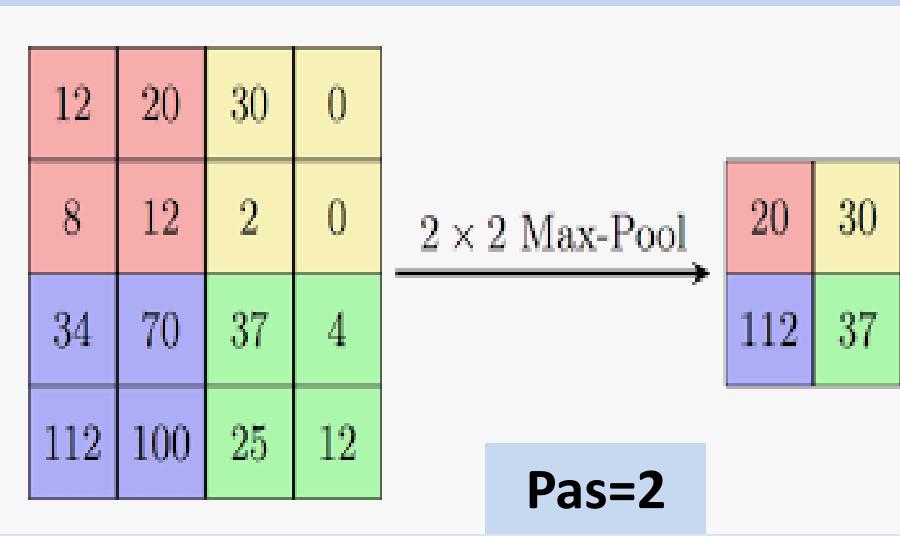
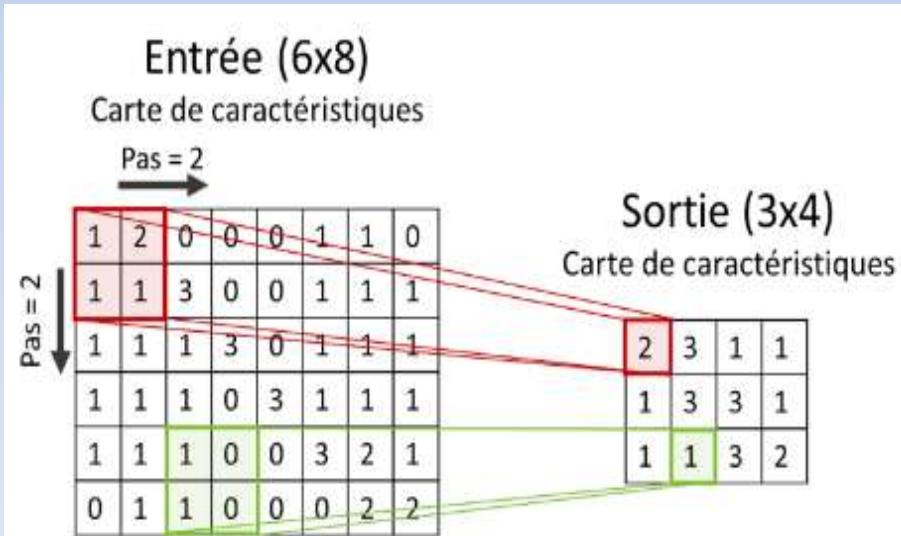
3. La **couche de regroupement** (mise en commun, pooling): réduit la dimension en remplaçant un sous-ensemble des entrées (sous-image) par une valeur, généralement le max.



Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

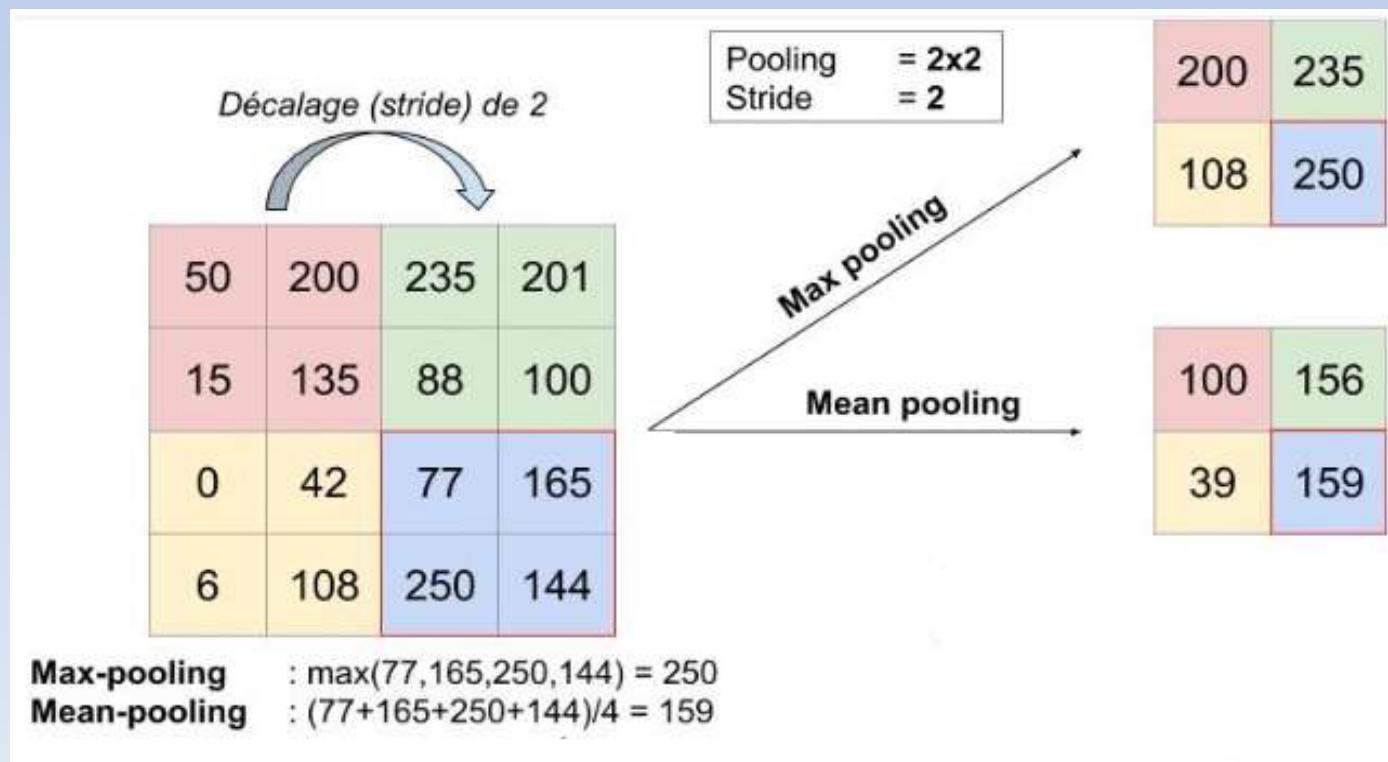
3. La couche de regroupement (mise en commun, pooling):



Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

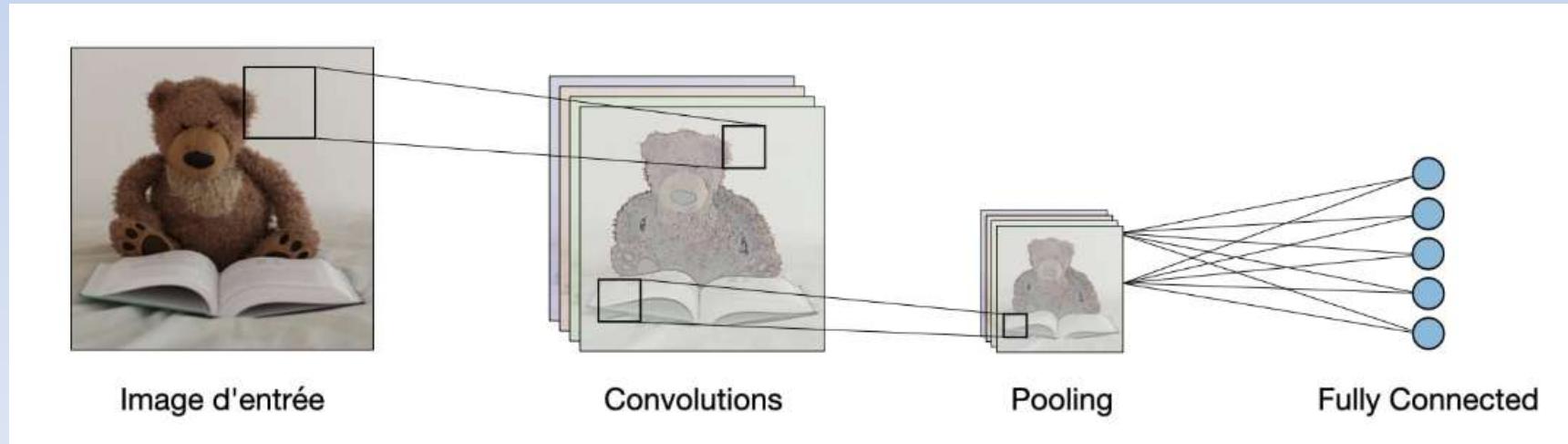
3. La couche de regroupement (mise en commun, pooling):



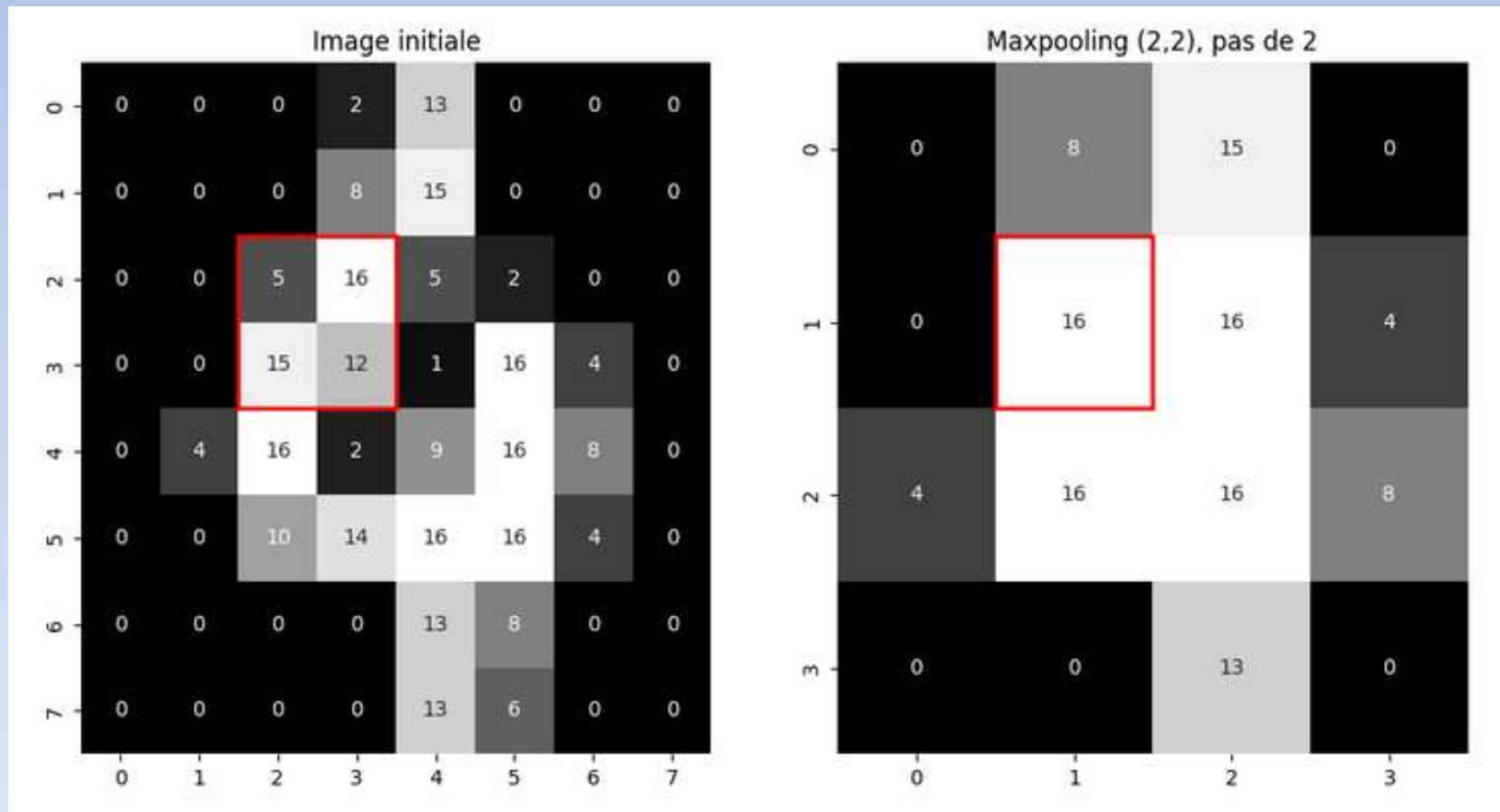
Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

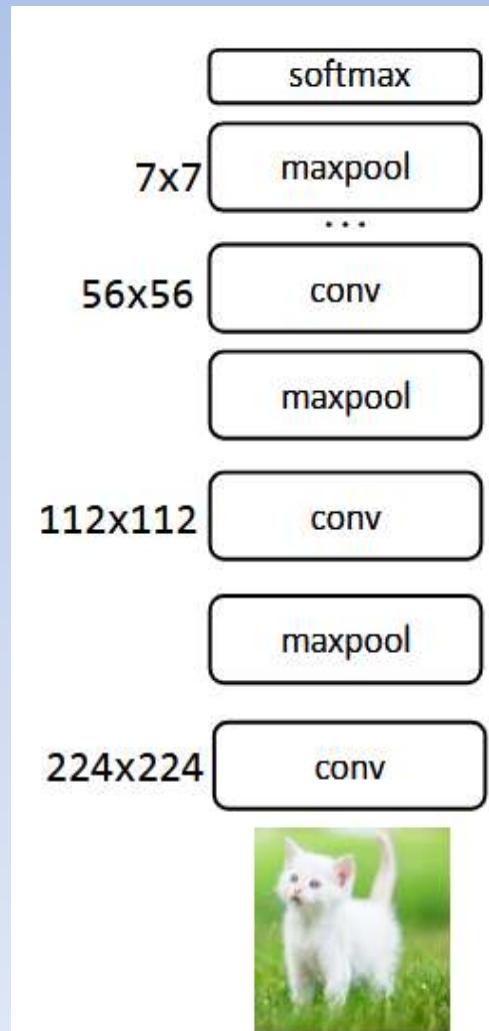
3. La couche de regroupement (mise en commun, pooling):



Réseau de neurones Convolutionnels



Réseau de neurones Convolutionnels

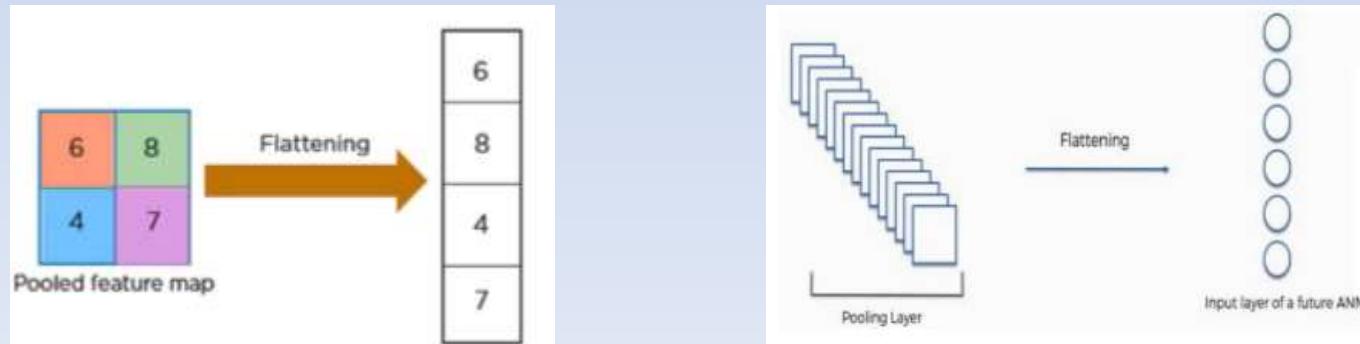


Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

4. Couche d'aplatissement (flattening)

L'aplatissement consiste à convertir tous les tableaux 2D en un seul vecteur linéaire vecteur linéaire continu

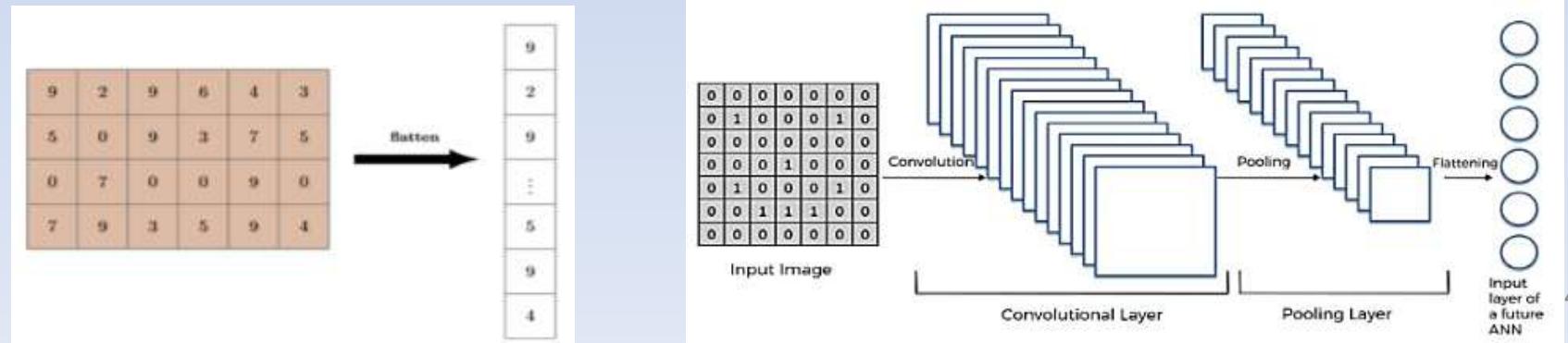


Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

4. Couche d'aplatissement (flattening)

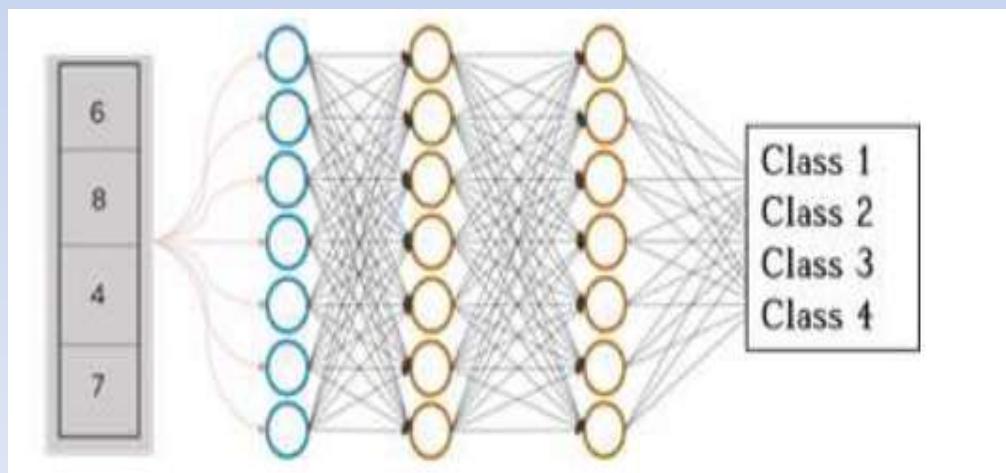
L'aplatissement consiste à convertir tous les tableaux bidimensionnels résultants en un seul vecteur linéaire vecteur linéaire continu



Réseau de neurones Convolutionnels

L'architecture de base du CNN comporte les couches :

5. La couche entièrement connectée: Elle permet de classifier l'image en entrée du réseau :



Réseau de neurones Convolutionnels

- Elle renvoie un vecteur de taille N , où N est le nombre de classes du problème traité.
- Chaque **élément** du vecteur indique **la probabilité** pour l'image en entrée d'appartenir à une classe.
- Elle applique une fonction d'activation (logistique si $N=2$, Softmax si $N>2$.)

Réseau de neurones Convolutionnels

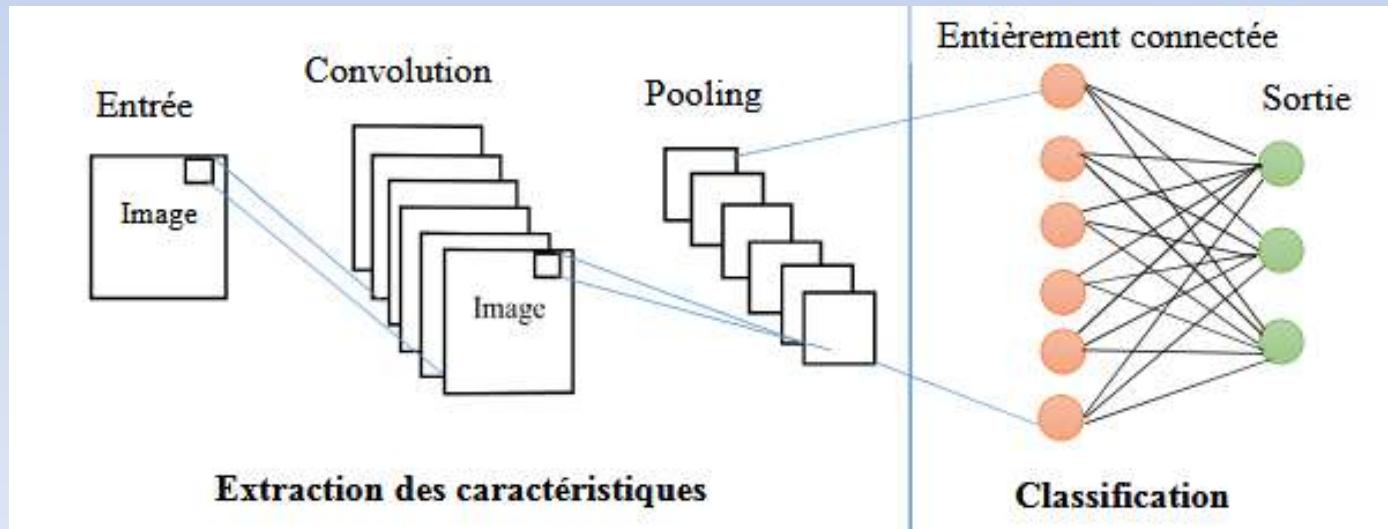
L'architecture de base du CNN comporte les couches :

5. Couche de perte (Loss):

- Elle mesure l'erreur du modèle.
- Elle est utilisée pour ajuster les poids et les paramètres du réseau lors de l'apprentissage, afin d'optimiser les performances du modèle.

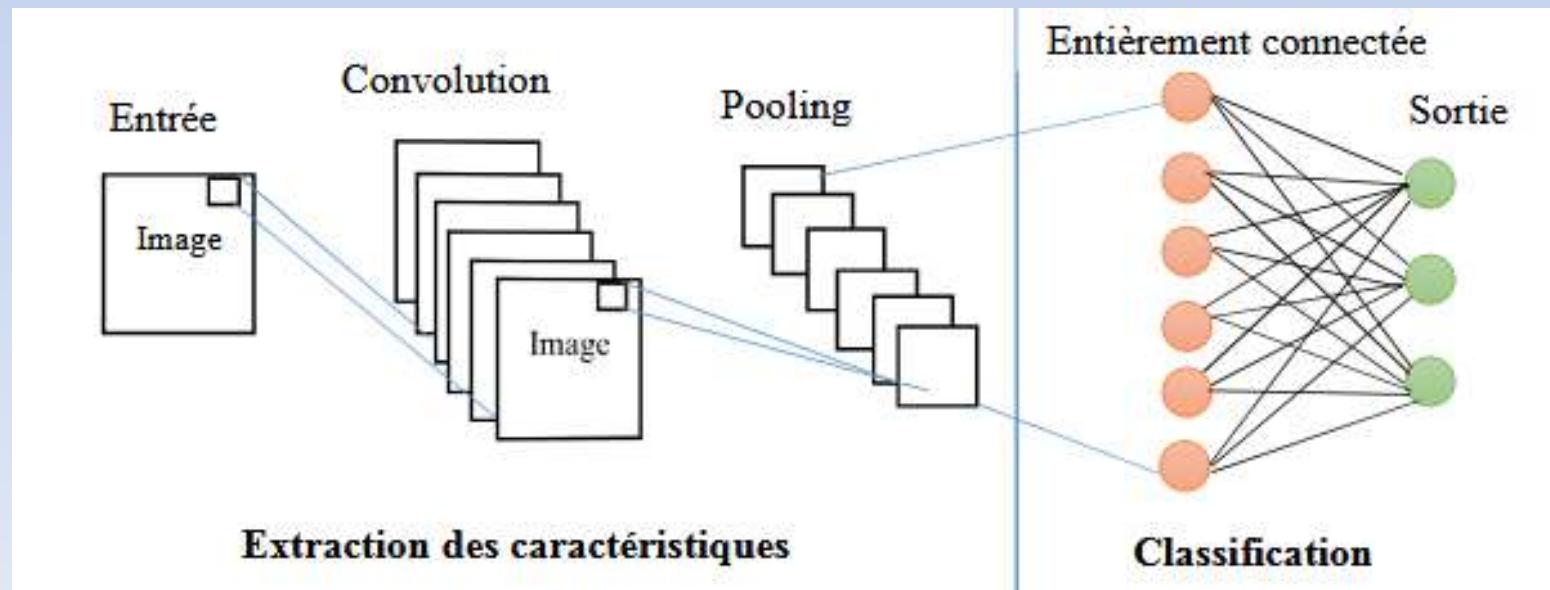
Réseau de neurones Convolutionnels

Cette **architecture** varie en fonction du problème à résoudre et devient un empilement de ces différentes couches.



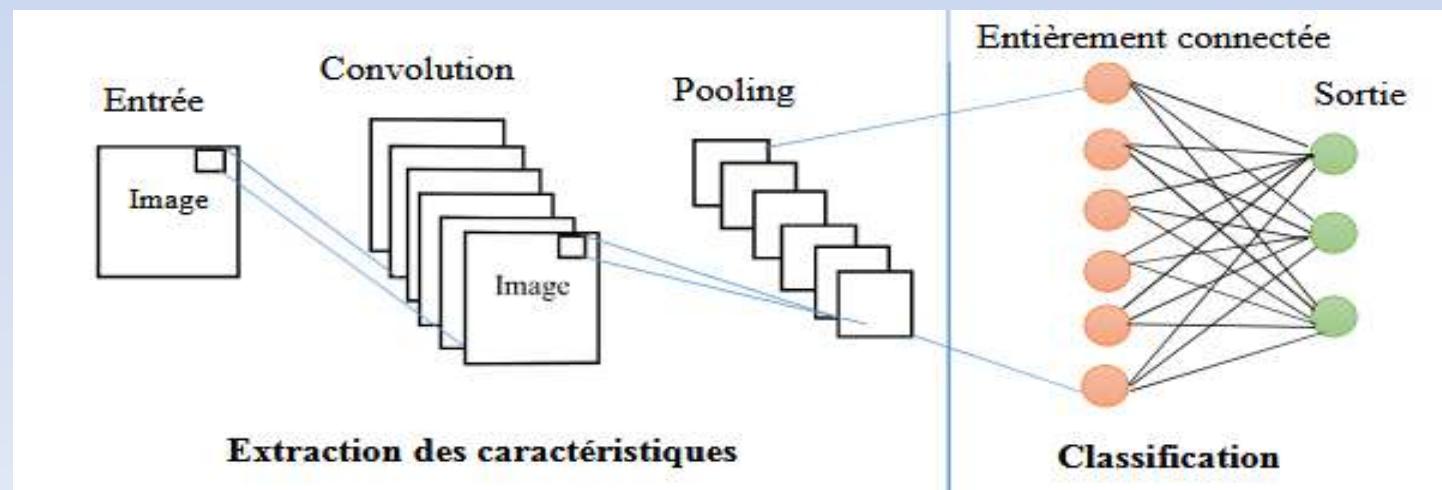
Réseau de neurones Convolutionnels

Tout CNN doit **commencer** par une couche de convolution et **terminer** par une couche entièrement connectée.



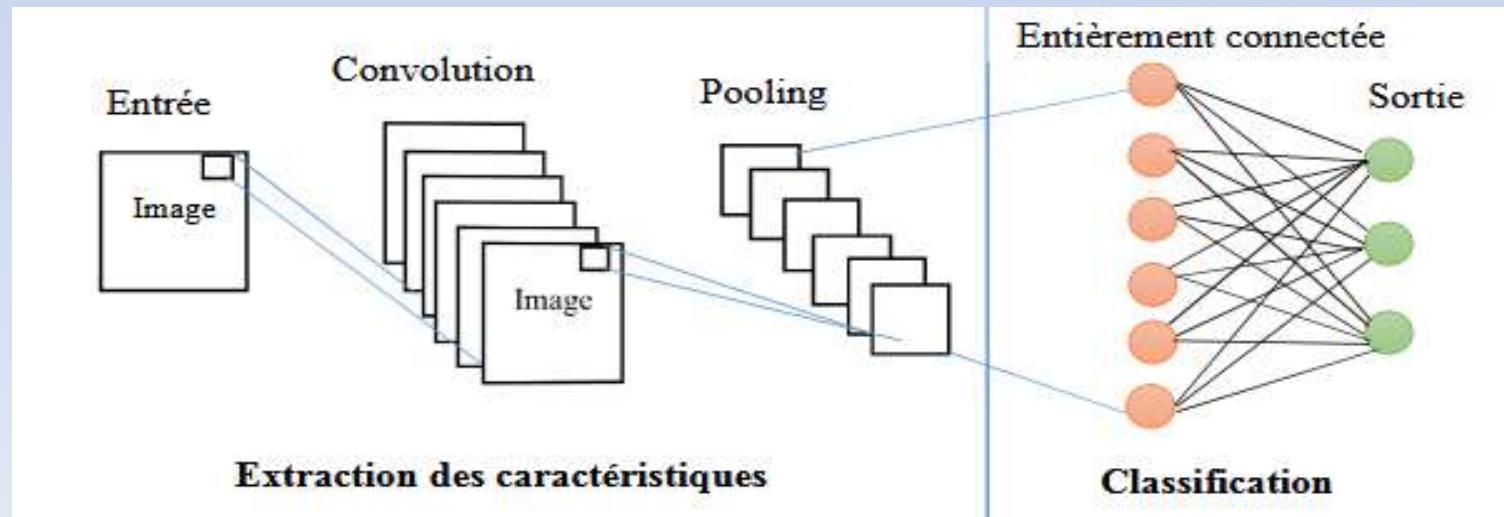
Réseau de neurones Convolutionnels

Les couches **intermédiaires** peuvent s'empiler de différentes manières, à condition que la sortie d'une couche ait la même structure que l'entrée de la suivante.



Réseau de neurones Convolutionnels

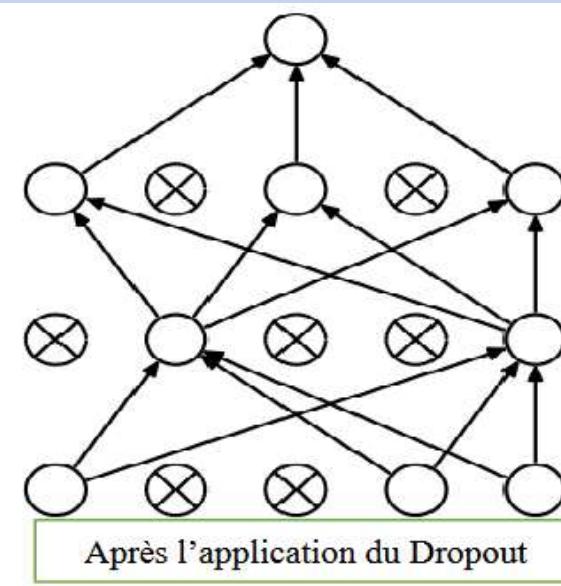
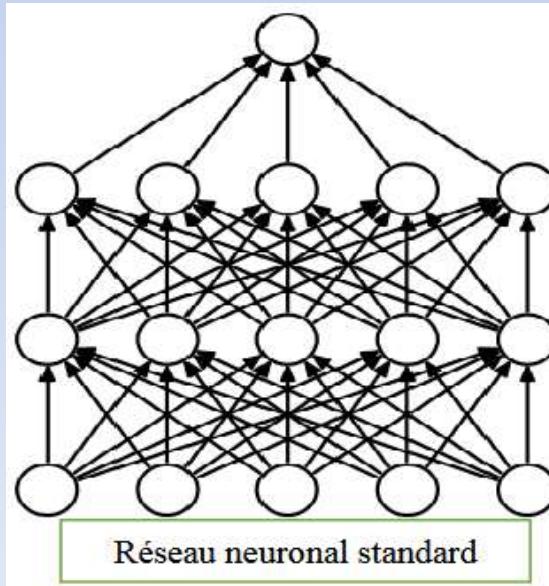
Par exemple, une couche **entièrement connectée**, qui renvoie toujours un vecteur, ne peut pas être placé avant une couche de pooling, puisque cette dernière doit recevoir une matrice.



Réseau de neurones Convolutionnels

Le dropout :

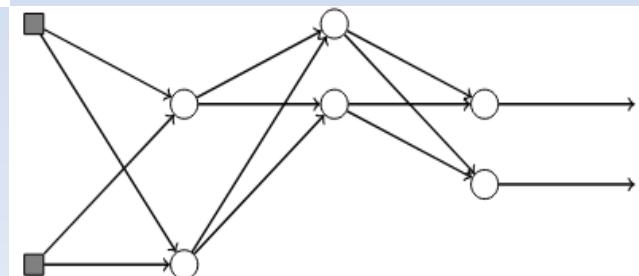
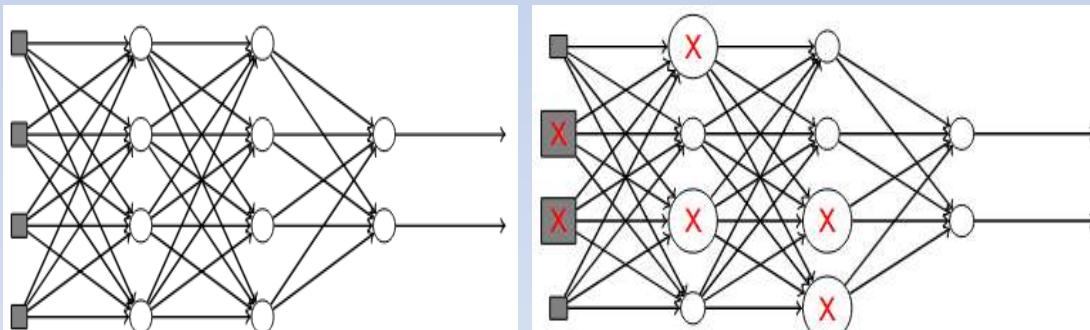
- Le Dropout est utilisés pour éviter le surapprentissage « l'overfitting » du réseau.



Réseau de neurones Convolutionnels

Le principe de cette technique est de:

Désactiver aléatoirement un certain pourcentage de neurones lors de l'entraînement du réseau.



Réseau de neurones Convolutionnels

Batch normalisation est:

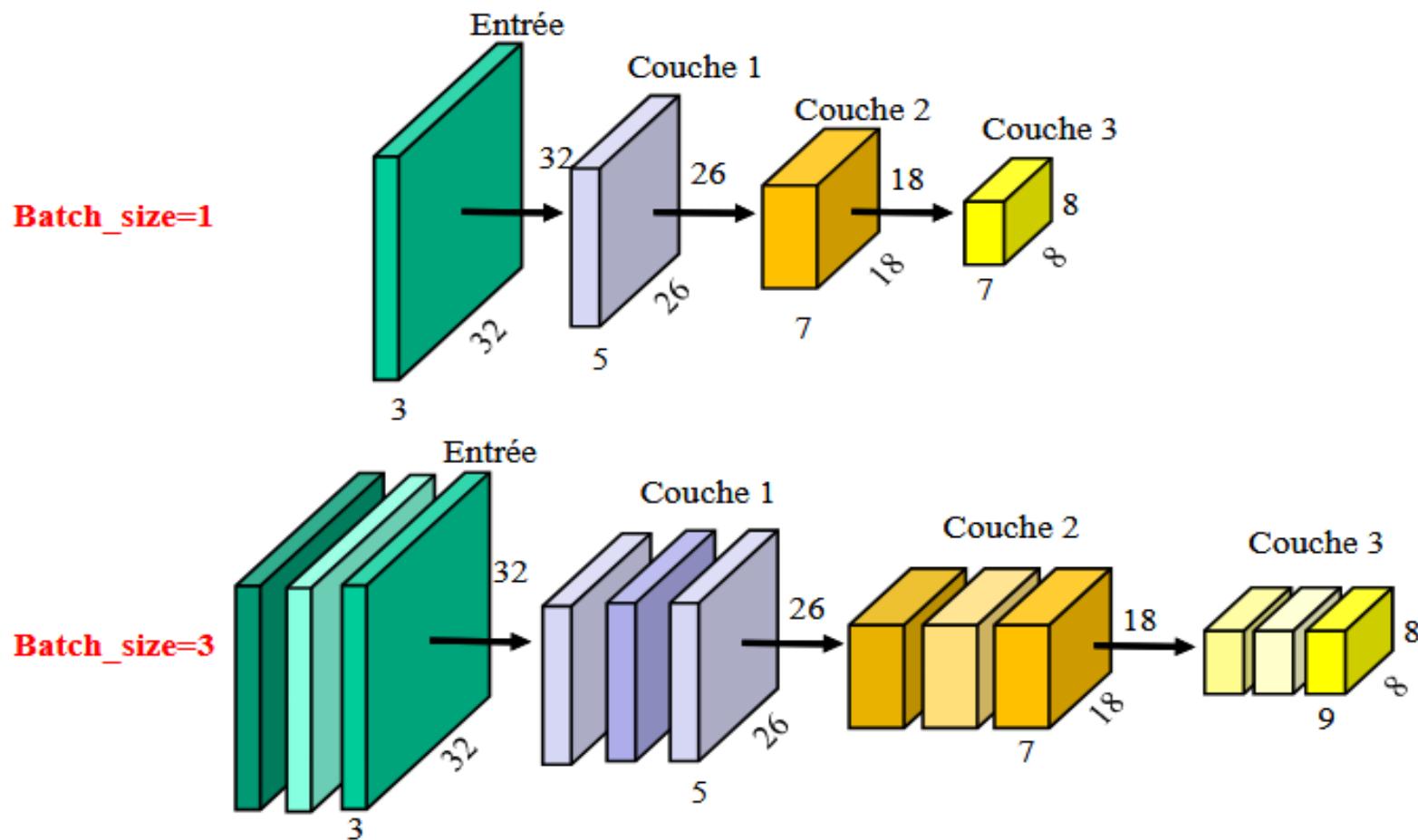
Une composante qui se trouve entre les couches du réseau de neurones.

Elle prend continuellement la sortie d'une couche particulière et la normalise avant de l'envoyer à la couche suivante comme entrée.

l'apprentissage du réseau converge plus rapidement si ses entrées sont **centrées réduites** (moyenne = 0, variance = 1).

Réseau de neurones Convolutionnels

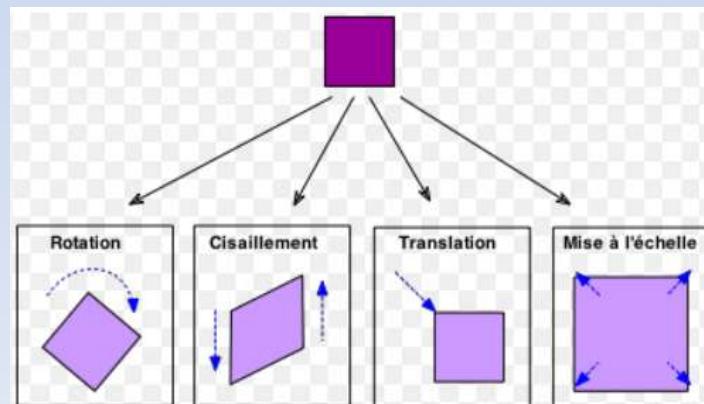
La taille du Batch (batch size)



Réseau de neurones Convolutionnels

Augmentation des données:

- Les techniques d'augmentation de données créées artificiellement différentes versions d'un jeu de données réelles pour augmenter sa taille.



Réseau de neurones Convolutionnels

Augmentation des données :

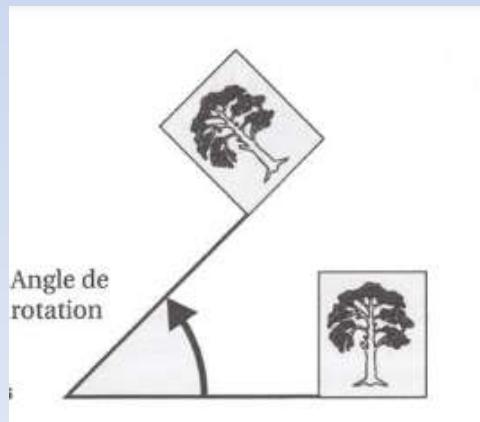
- Les algorithmes d'augmentation de données peuvent augmenter la précision des modèles d'apprentissage automatique

Réseau de neurones Convolutionnels

Transformations géométriques

- **Rotation**

Cette transformation, comme son nom l'indique, consiste à effectuer une rotation de l'image originale selon un angle souhaité.

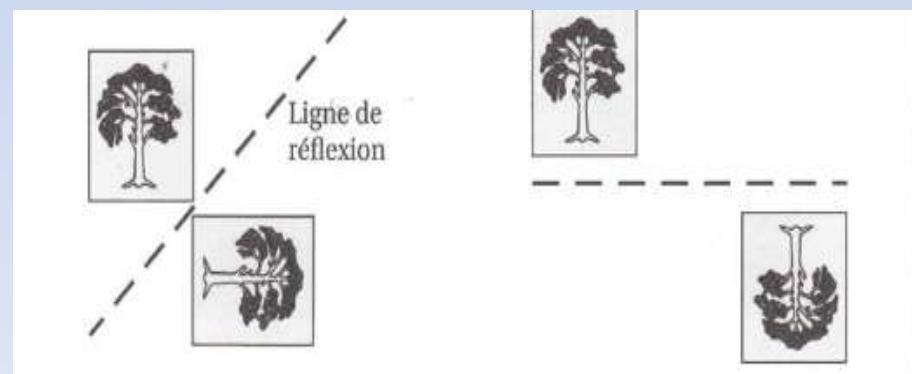
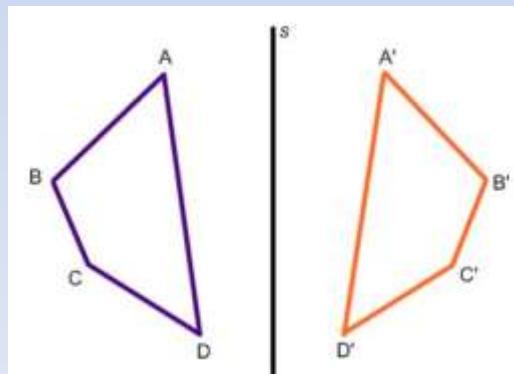


Réseau de neurones Convolutionnels

Transformations géométriques

- **Réflexions (en anglais flips)**

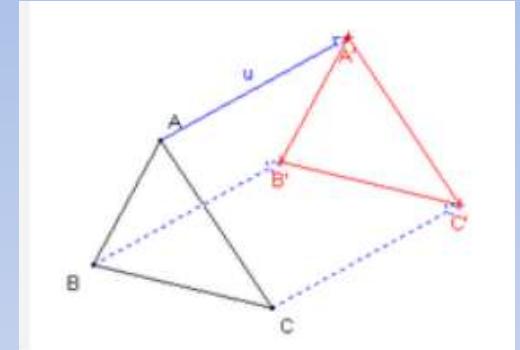
Les réflexions d'images s'effectuent selon un axe de symétrie.



Réseau de neurones Convolutionnels

Transformations géométriques

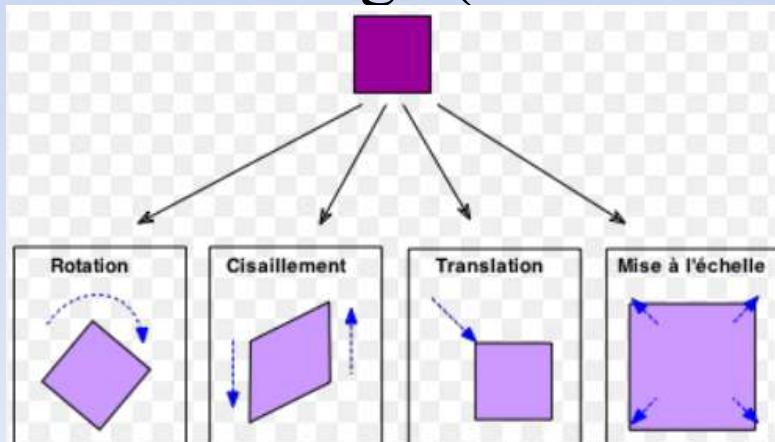
- **Translations (en anglais shift)**
 - Cette transformation peut être effectuée selon l'axe x et/ou y.
 - L'image transformée conserve la même orientation que l'image originale et est déplacée selon la direction appliquée.



Réseau de neurones Convolutionnels

Transformations géométriques

- **Cisaillement** (en anglais **shear**): une transformation affine qui consiste à décaler dans des directions opposées le haut et le bas de l'image (cisaillement horizontal) ou la droite et la gauche de l'image (cisaillement vertical).

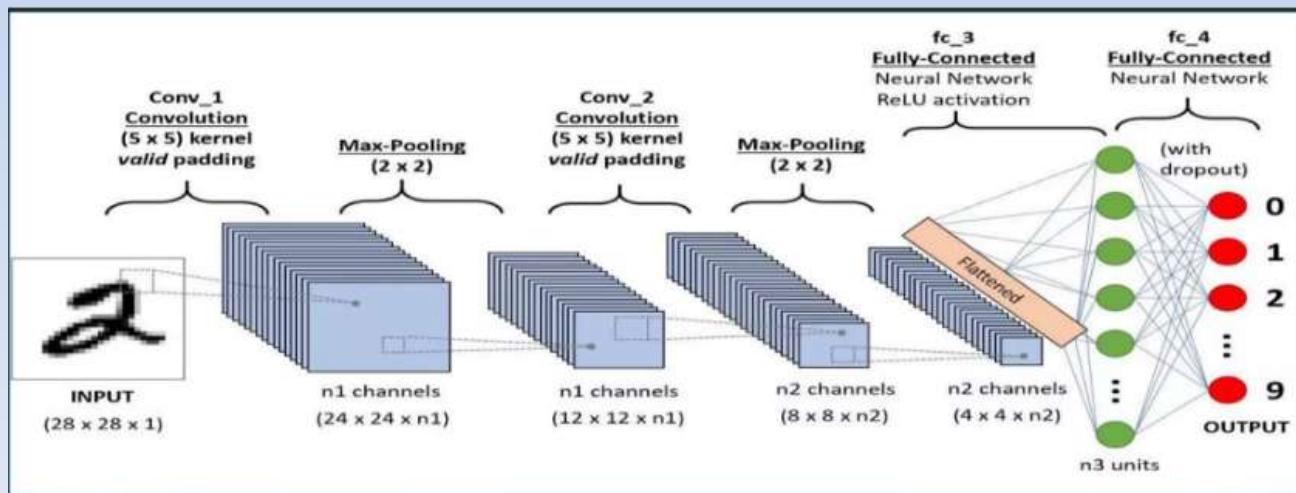


Réseau de neurones Convolutionnels

Paramétrage des couches

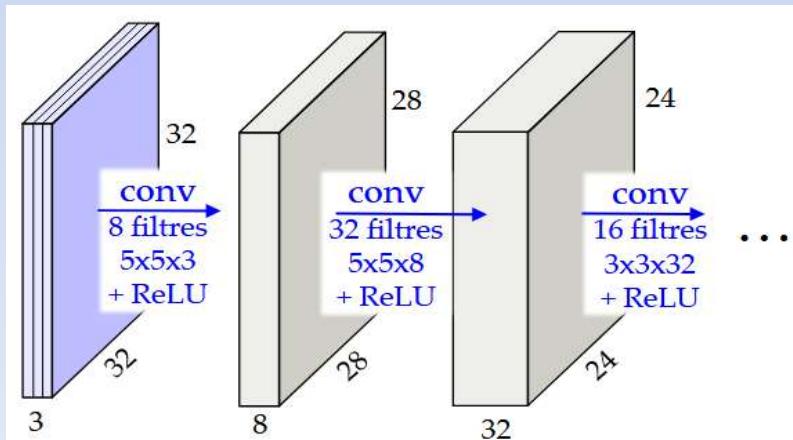
Paramétrage des couches

- La couche de convolution possède **quatre** hyper-paramètres :
 1. Le nombre de filtres **K**.



Paramétrage des couches

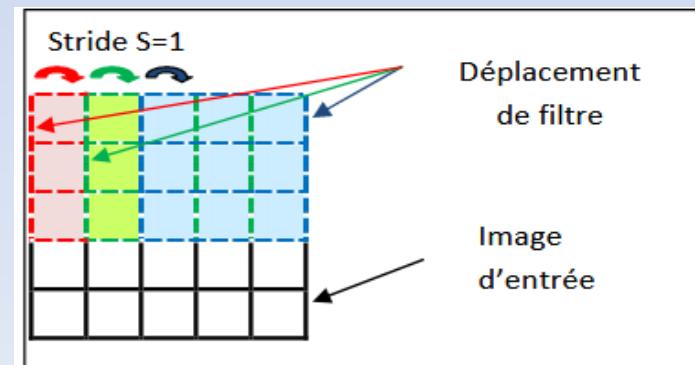
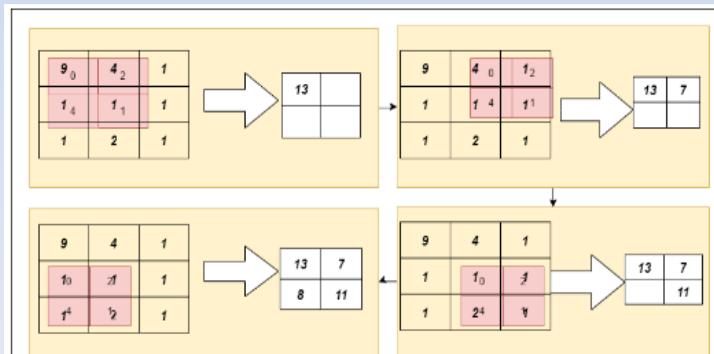
- La couche de convolution possède **quatre** hyperparamètres :
 2. La taille F des filtres : chaque filtre est de dimensions $F \times F \times D$ pixels.
- D: Profondeur de l'image



Paramétrage des couches

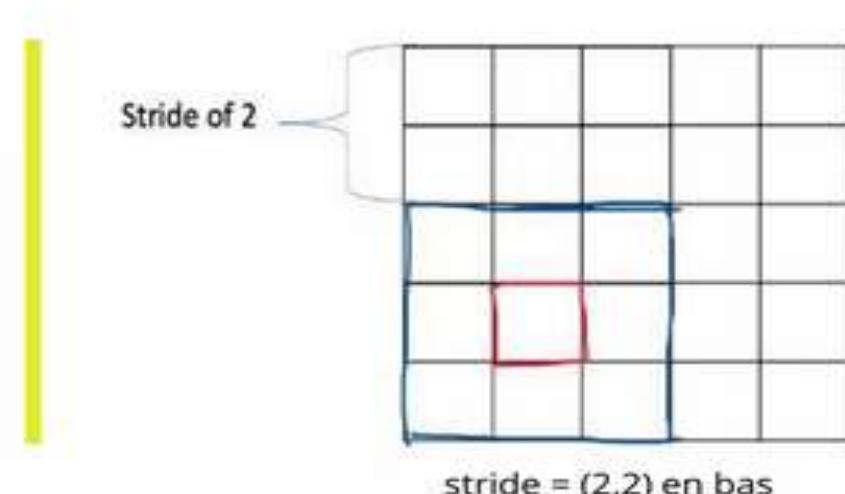
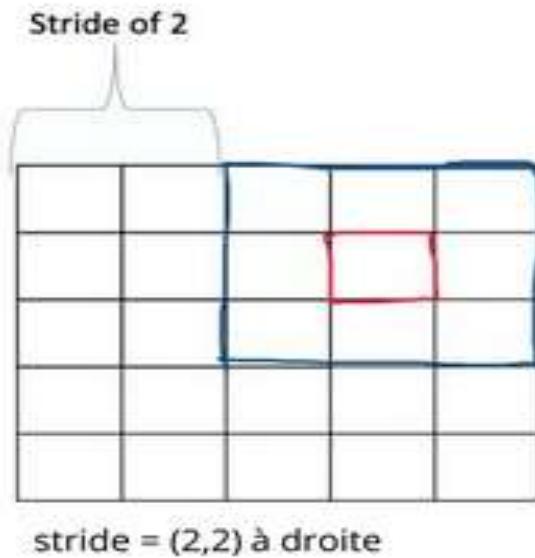
3. Le pas **S (stride)** avec lequel on fait glisser la fenêtre correspondant au filtre sur l'image.

Par exemple, un pas de 1 signifie qu'on déplace la fenêtre (filtre) d'un pixel à la fois.



Paramétrage des couches

Exemple de Stride



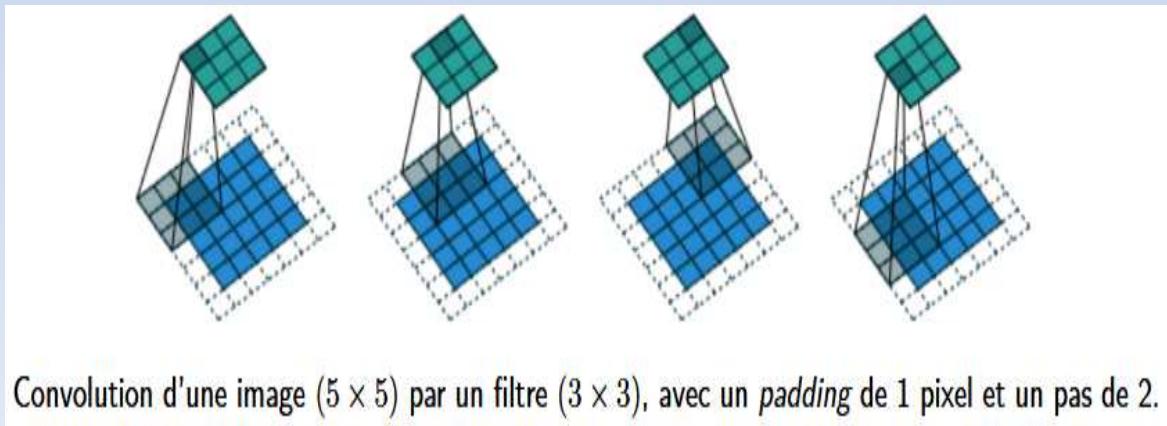
Paramétrage des couches

La couche de convolution possède 4 hyper-paramètres :

4. Le remplissage (*padding*) \mathbf{P} :

Pour un filtre \mathbf{h} de dimensions (w, h) , que faire le long des bords de l'image?

Il y a $\mathbf{h}/2$ (ou $w/2$) pixels qui sont dans le filtre mais hors de l'image.

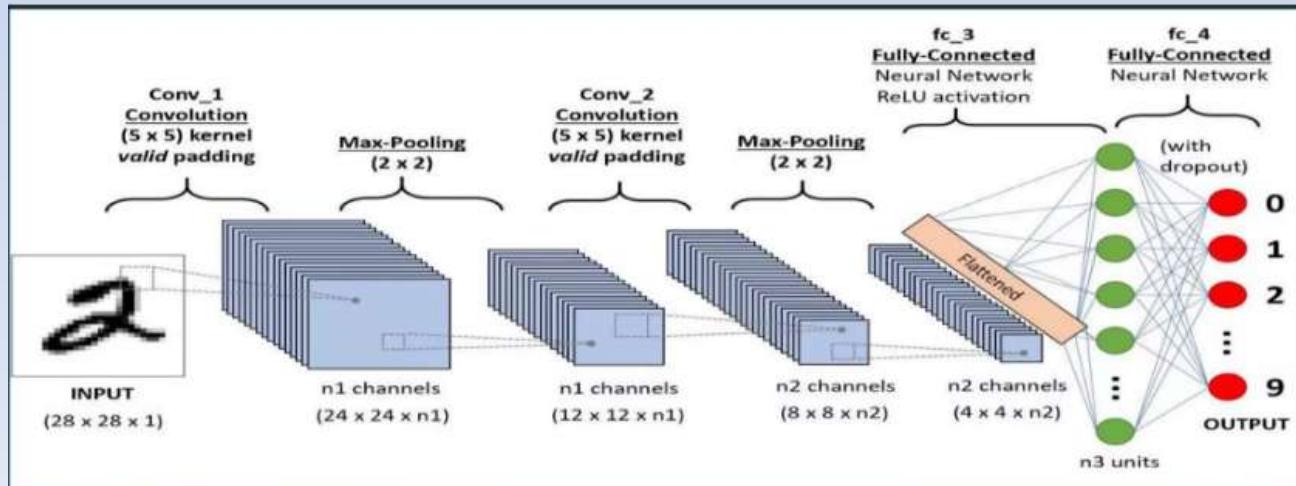


Convolution d'une image (5×5) par un filtre (3×3) , avec un *padding* de 1 pixel et un pas de 2.

Paramétrage des couches

Option 1: remplissage « valide » (valid padding)

- ⇒ Aucun remplissage n'est ajouté à l'entrée.
- ⇒ On ne calcule pas la convolution sur ces pixels.
- ⇒ La sortie est **plus petite** que l'entrée.



Paramétrage des couches

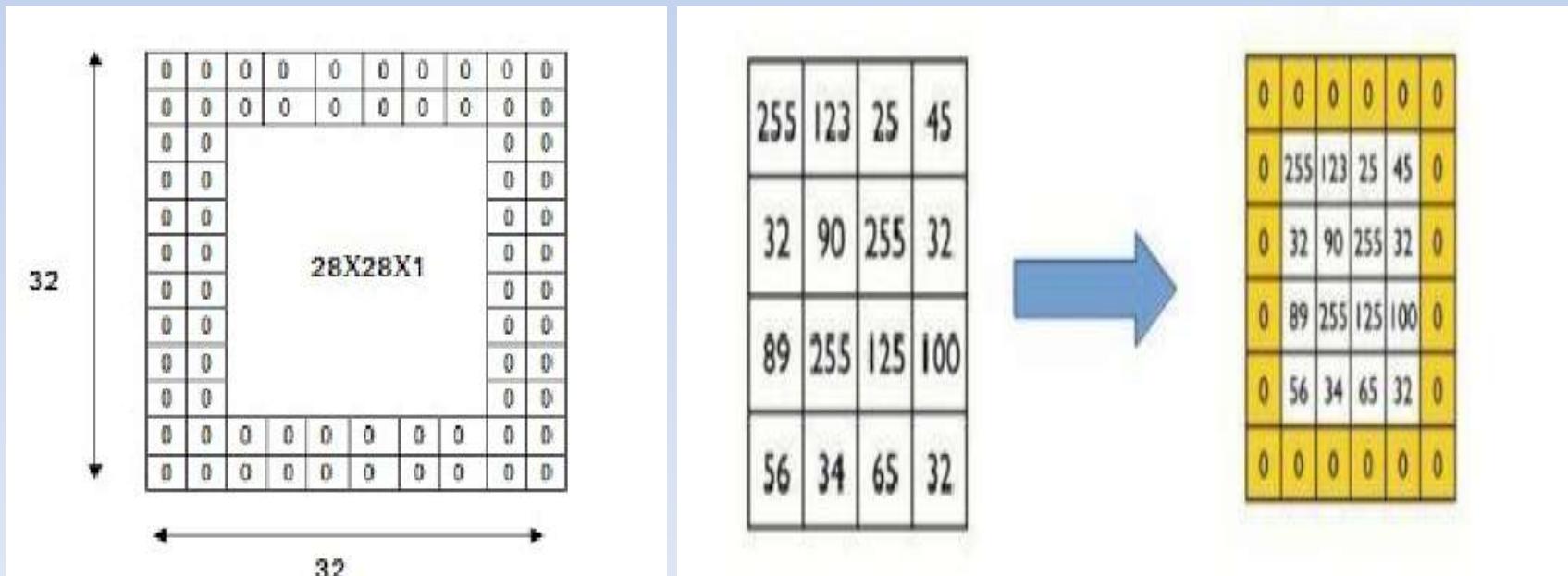
Option 2: remplissage « identique »:

- ⇒ Ajouter des pixels autour de l'entrée.
 - ⇒ La taille de l'entrée après convolution est identique à celle avant convolution.
 - ⇒ Les valeurs utilisées: zéro, recopie, symétrie, etc.

Paramétrage des couches

4. Le zeros *padding* P : consiste à ajouter P zéros à chaque côté des frontières de l'entrée.

* Le résultat possède la même taille que l'entrée.



Architectures des RN

Réseau de neurones convolutionnels

La couche convolution: Pour chaque image de taille $W_1 \times H_1 \times D_1$ en entrée, la couche de convolution renvoie une matrice de dimensions $W_2 * H_2 * D_2$, où

$$W_2 = (W_1 - F + 2P)/S + 1$$

F: taille du filtre

S: stride

P: padding

$$H_2 = (H_1 - F + 2P)/S + 1$$

$$D_2 = K$$

K: le nombre de filtres appliqués

Architectures des RN

Tailles entrée et sortie

K : nombre de filtres

F : taille du filtre (FxF)

S : pas (stride)

P : quantité de 0-padding

Valeurs typiques (cs231n)

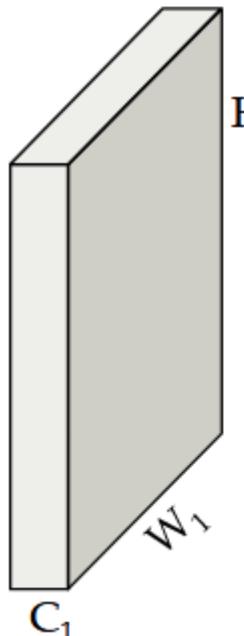
K : puissances de 2

F=3, S=1, P=1

F=5, S=1, P=2

F=5, S=2, P=autant que nécessaire

F=1, S=1, P=0



$$H_2 = \frac{H_1 - F + 2P}{S} + 1$$

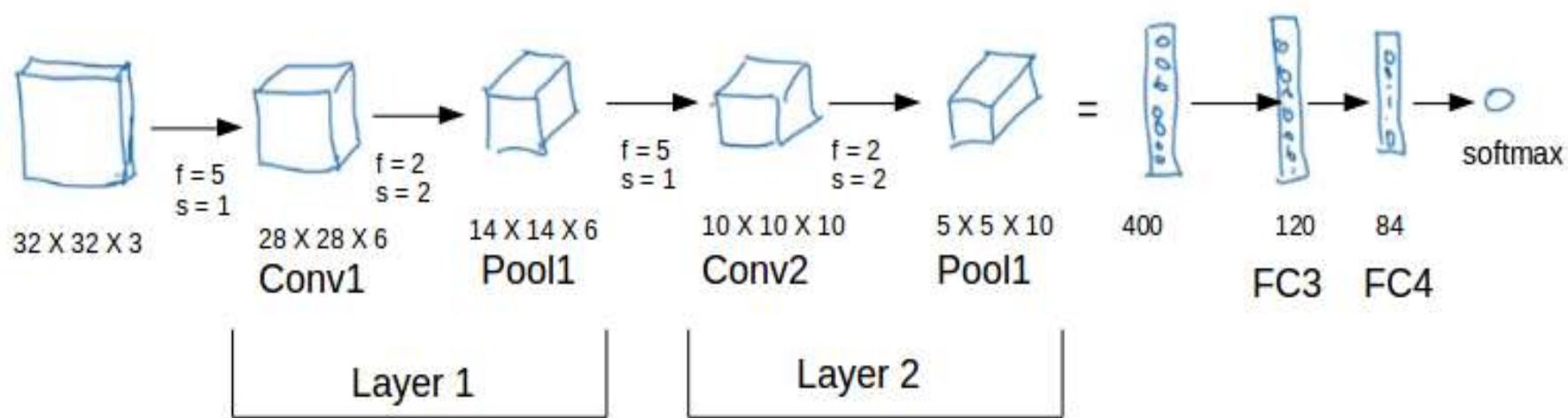
$$W_2 = \frac{W_1 - F + 2P}{S} + 1$$

Architectures des RN

Réseau de neurones convolutionnels

Exemple:

$$W2 = (W1 - F + 2P) / S + 1$$



Architectures des RN

Réseau de neurones convolutionnels

La couche de pooling présente seulement 2 hyperparamètres :

1. La taille **F** de la fenêtre: l'image est découpée en cellules carrées de taille $F \times F$ pixels
2. Le pas **S** : les cellules sont séparées les unes des autres de S pixels

Architectures des RN

Réseau de neurones convolutionnels

La couche de pooling:

Pour chaque image de taille $W \times H \times D$ en entrée, la couche de pooling renvoie une matrice de dimensions :

$$W_p * H_p * D_p,$$

$$W_p = (W - F) / S + 1$$

$$H_p = (H - F) / S + 1$$

$$D_p = D$$

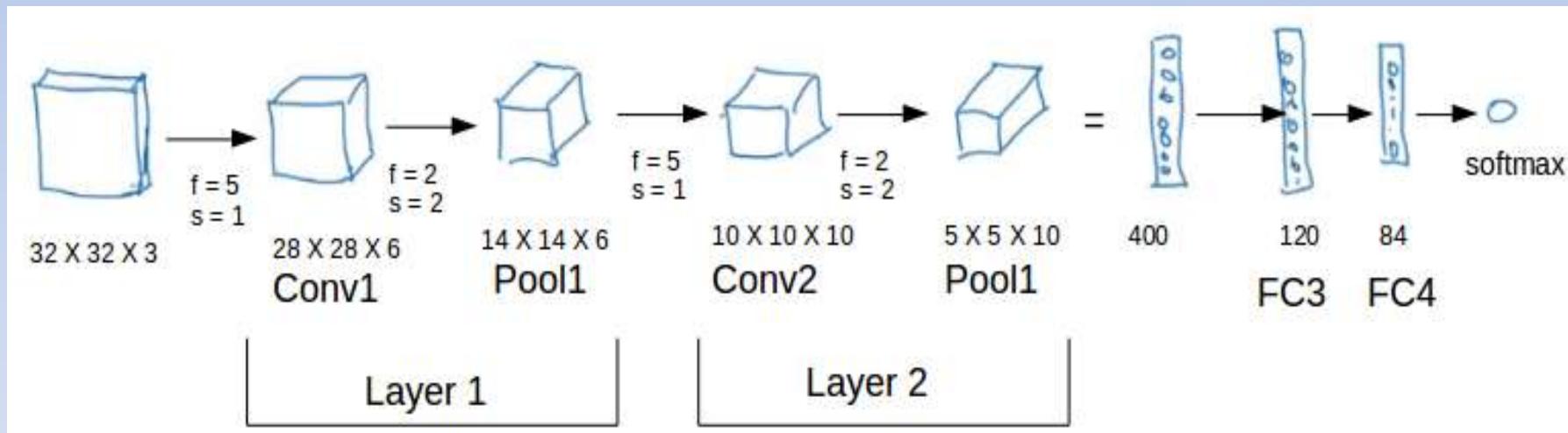
F: taille du filtre

S: stride

Architectures des RN

Réseau de neurones convolutionnels

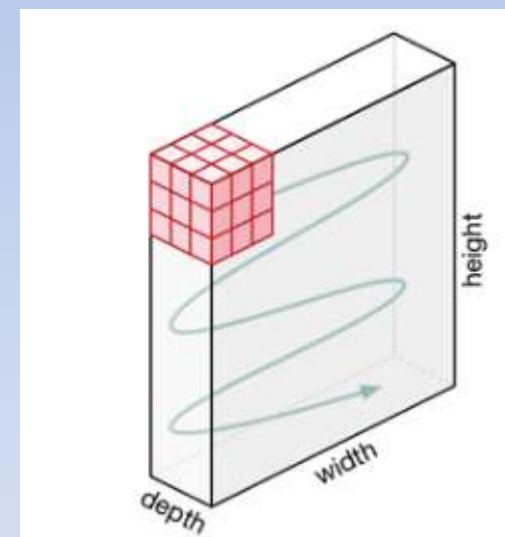
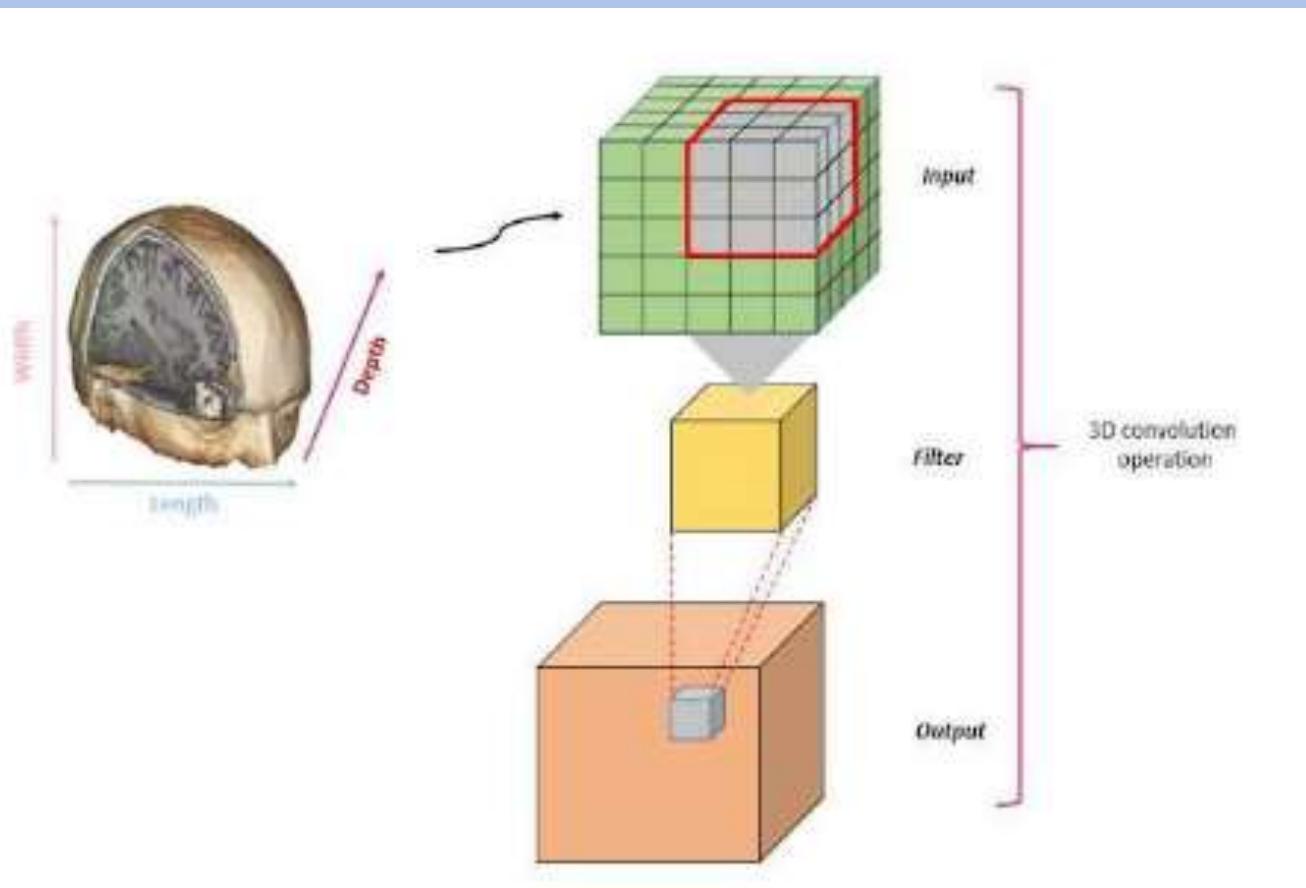
Exemple: $Wp = (W-F)/S + 1$





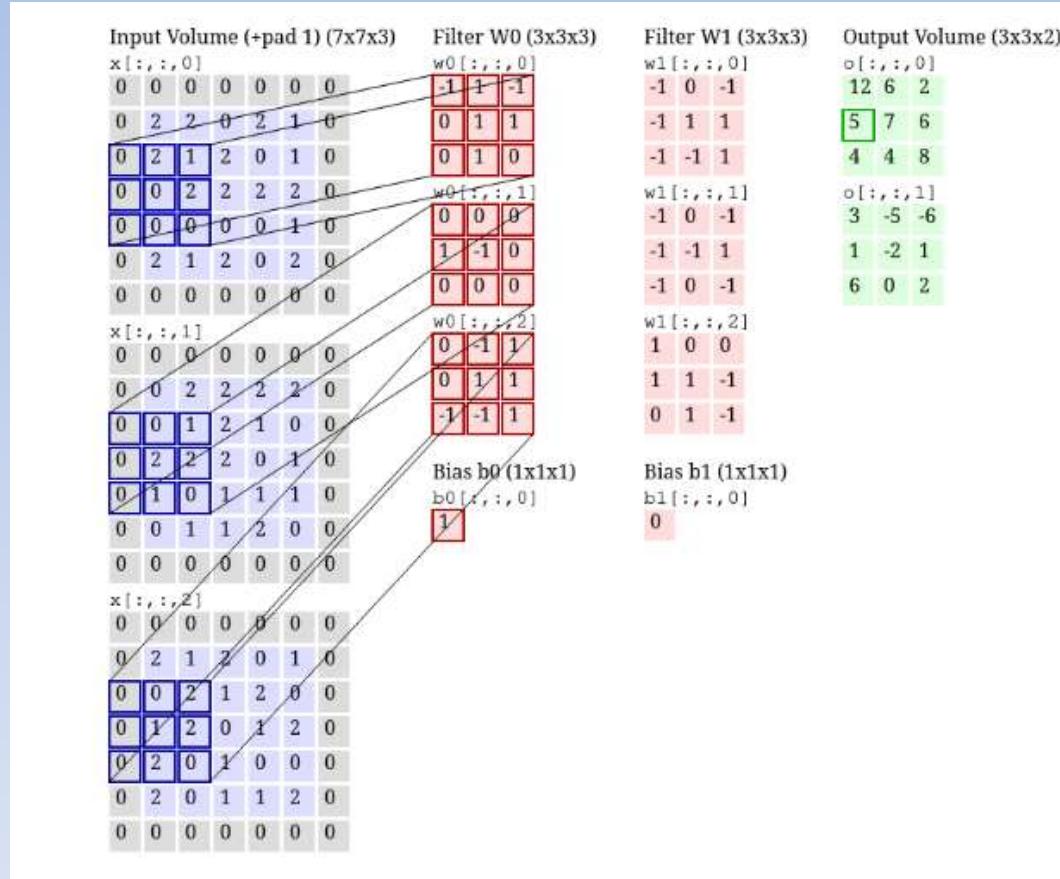
CNN 3D

Architectures d'un CNN 3D



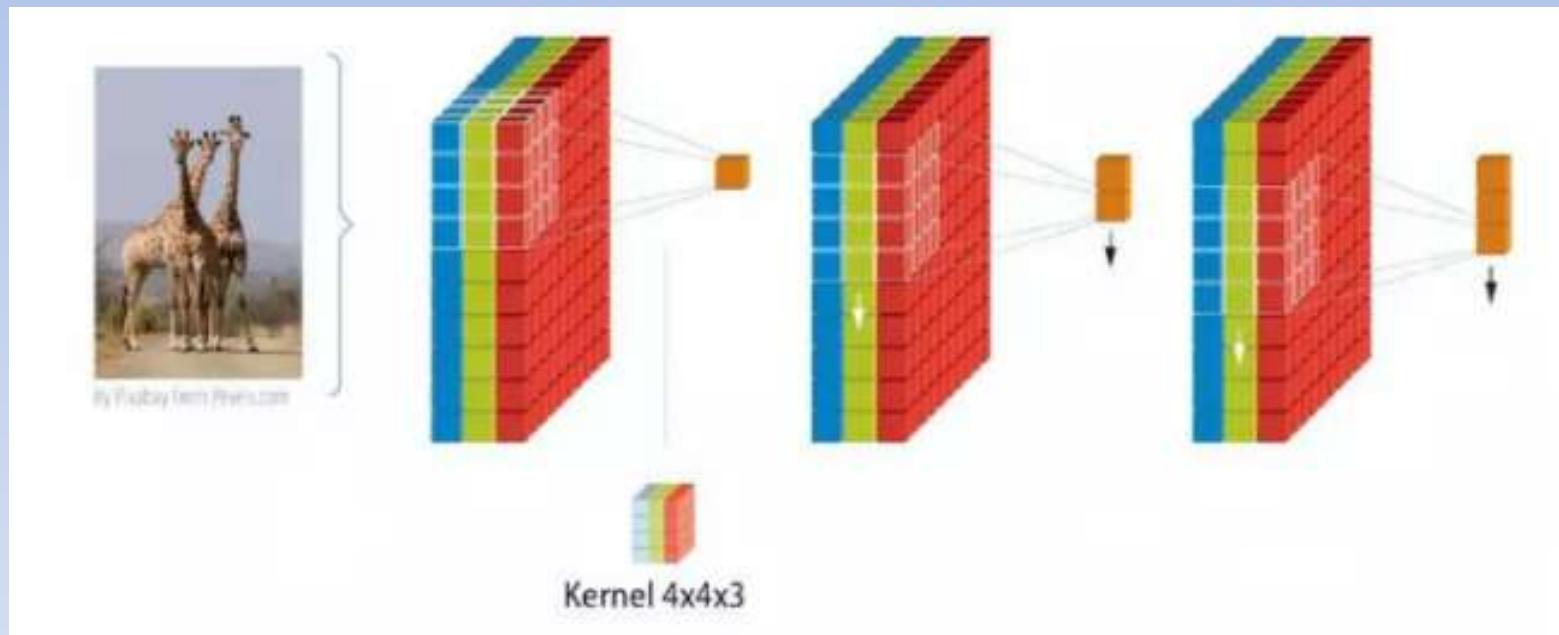
La convolution 3D

Architectures d'un CNN 3D



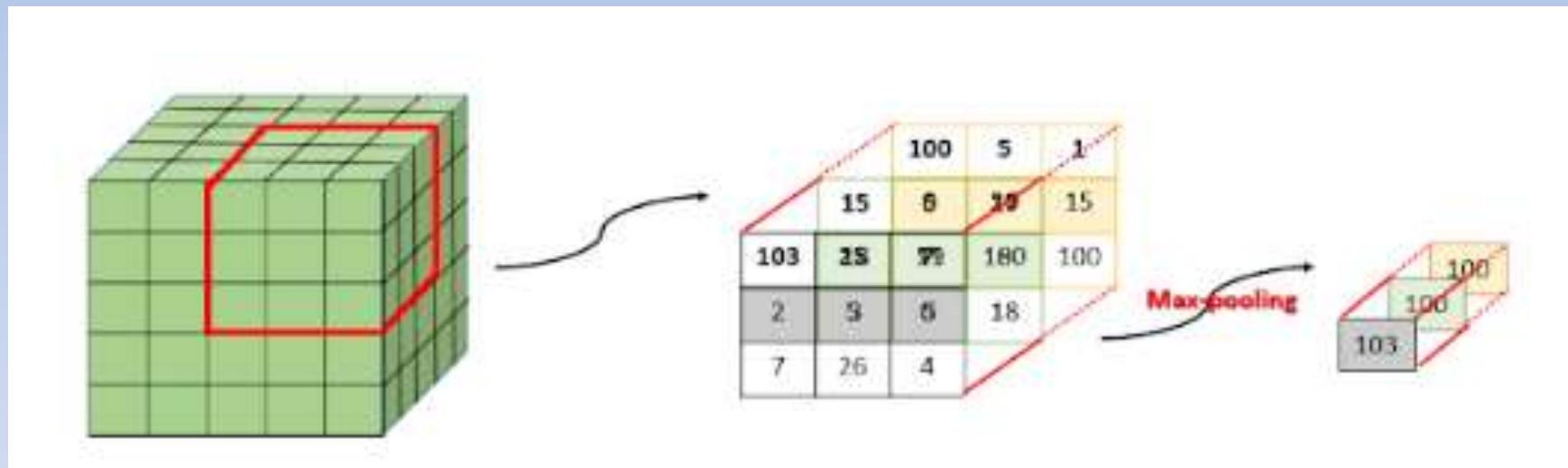
La convolution 3D

Architectures d'un CNN 3D



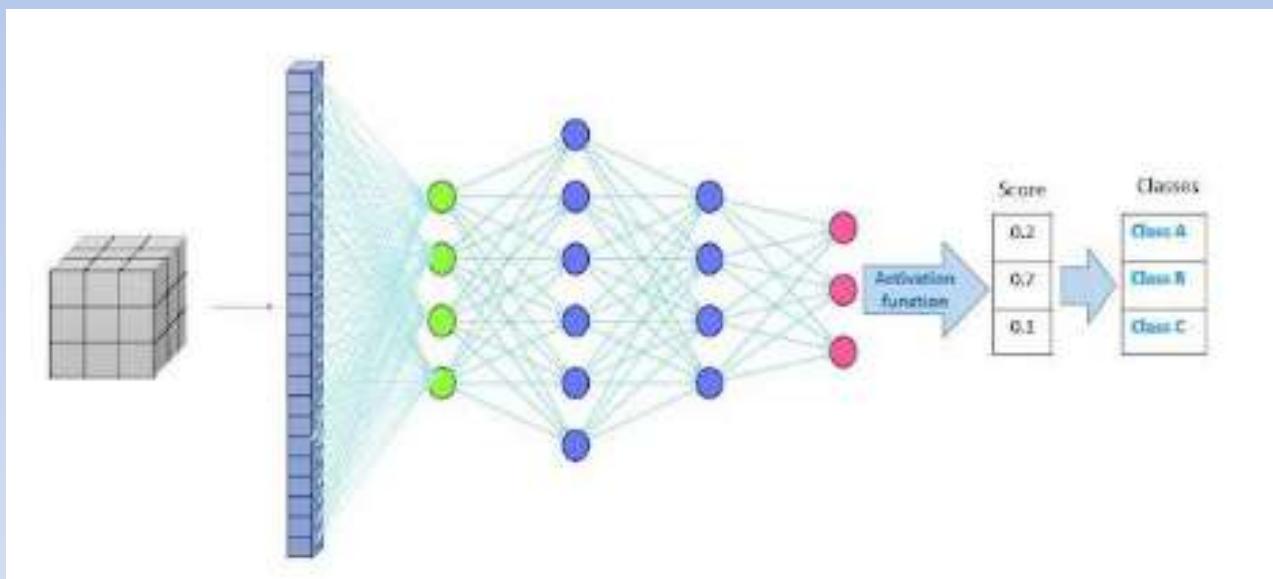
La convolution 3D

Architectures d'un CNN 3D



Pooling 3D

Architectures d'un CNN 3D



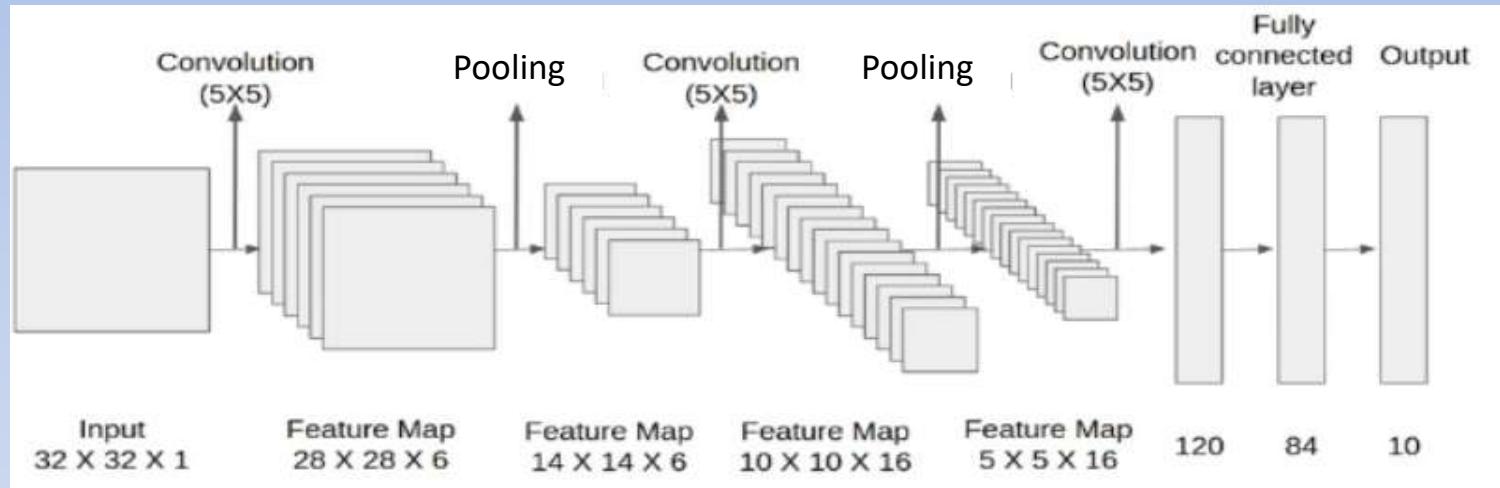
Couche entièrement connectée

Exemples de CNN

LeNet-5

Exemples de CNN

Architecture du réseau LeNet-5



Il comprend au total **5 couches cachées** :

- **Convolution 5*5:** $32*32*1 \Rightarrow 28*28*6$
- **Convolution 5*5:** $14*14*6 \Rightarrow 10*10*6$
- **Convolution 5*5:** $5*5*16 \Rightarrow 1*1*120$
- **Couche entièrement connectée:** $120 \Rightarrow 84$
- **Couche entièrement connectée:** $84 \Rightarrow 10$

$$W_2 = (W_1 - F + 2P) / S + 1$$

AlexNet

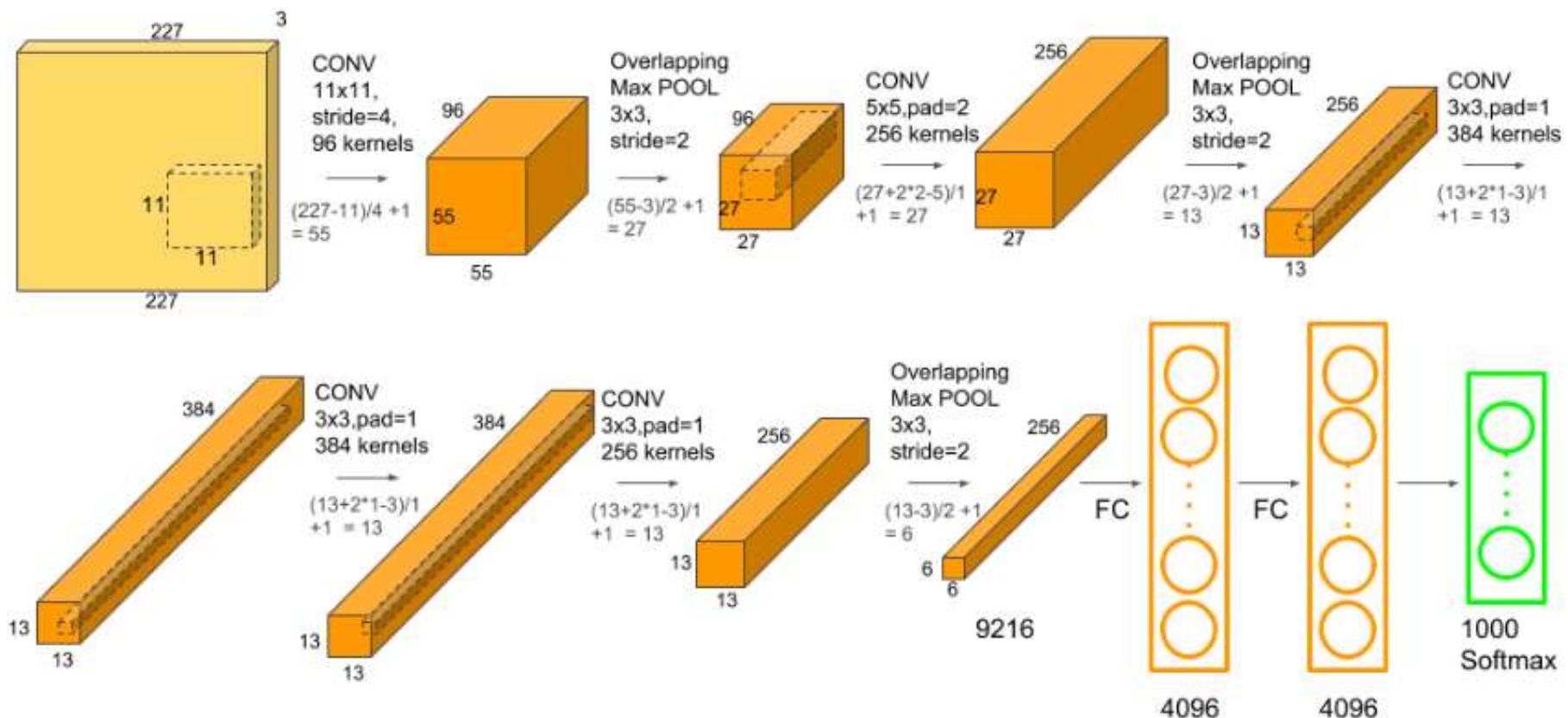
AlexNet

- Il est composé de **5** couches de convolution.
- **3** couches entièrement connectés.
- Il utilise des techniques telles que l'utilisation de: couches de Pooling, de dropout et de normalisation.

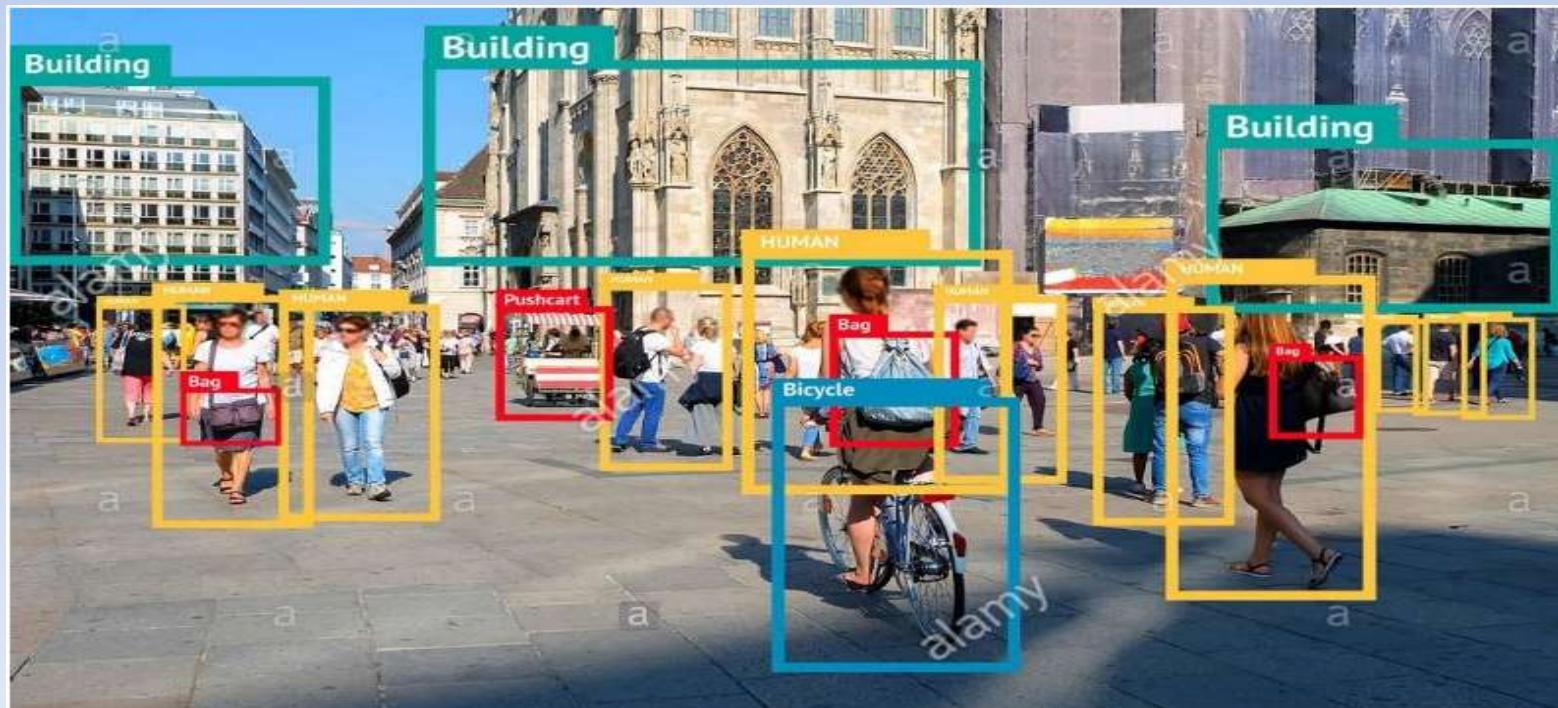
AlexNet

Architecture du réseau AlexNet

$$W_2 = (W_1 - F + 2P)/S + 1$$

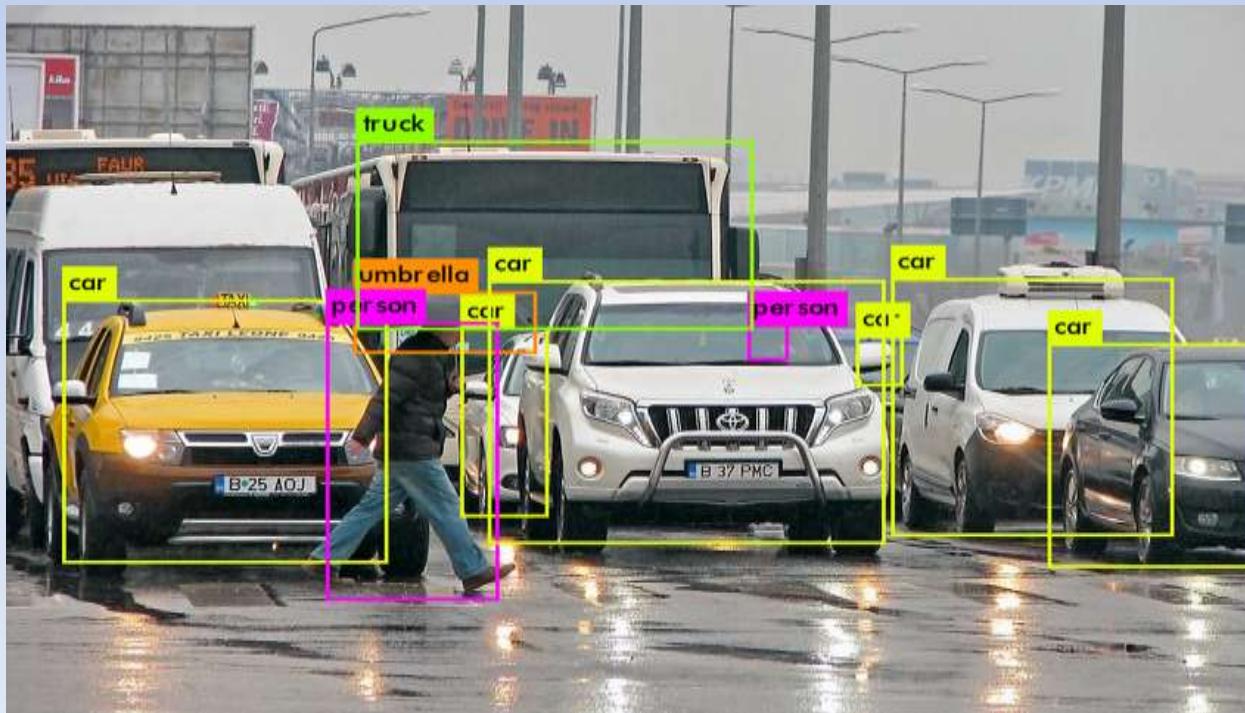


YOLO (You Only Look Once)



YOLO

La **détection d'objets** est une branche de la vision par ordinateur qui identifie et localise des objets dans une image ou une vidéo.

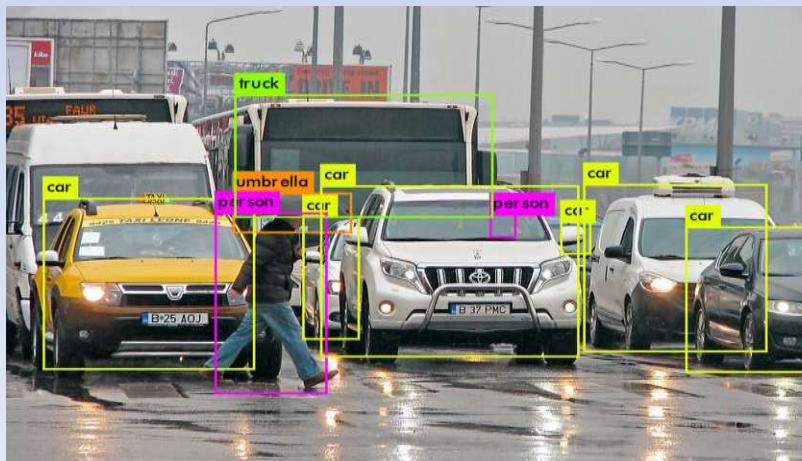


YOLO (You Only Look Once)

La détection d'objets cherche à répondre à deux questions fondamentales :

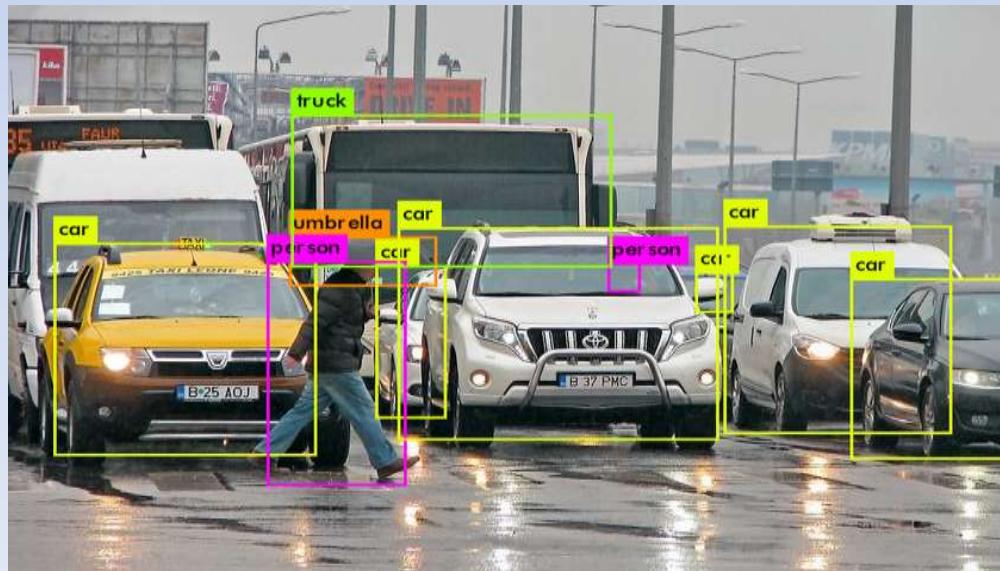
Qu'est-ce que l'objet ? Cette question vise à identifier l'objet dans une image spécifique.

Où est-il? Cette question vise à trouver l'emplacement exact de l'objet dans l'image.



YOLO (You Only Look Once)

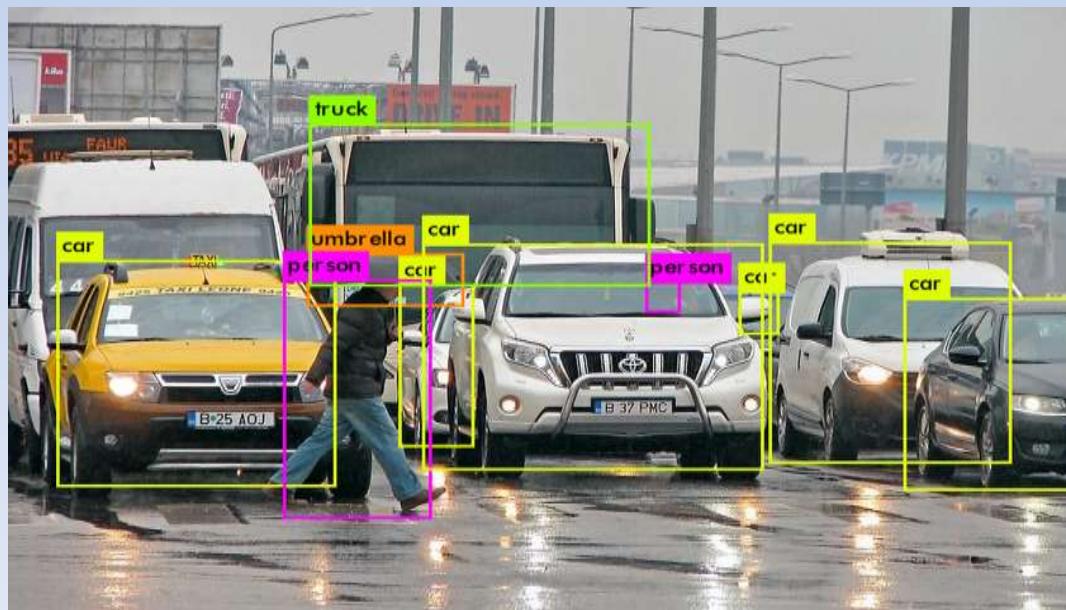
- L'image n'est pas contrainte d'avoir un **seul** objet, mais peut contenir **plusieurs** objets.
- La tâche est de classer et de localiser tous les objets dans l'image.



YOLO (You Only Look Once)

La localisation est faite:

- En utilisant le concept de la **boîte de délimitation** (**Anchor box**) qui peut être identifiée par certains paramètres numériques tout en respectant la limite de l'image.



YOLO (You Only Look Once)

1. Préparer l'image en entrée:

Tout d'abord, l'algorithme YOLO commence par une image.

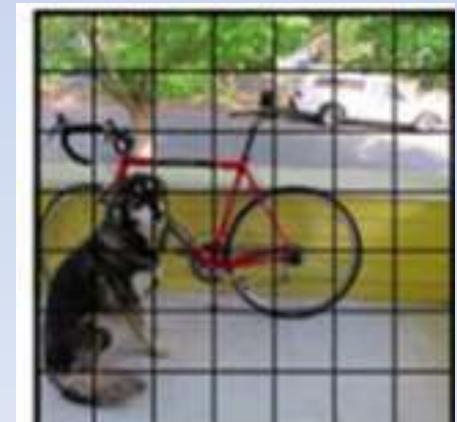
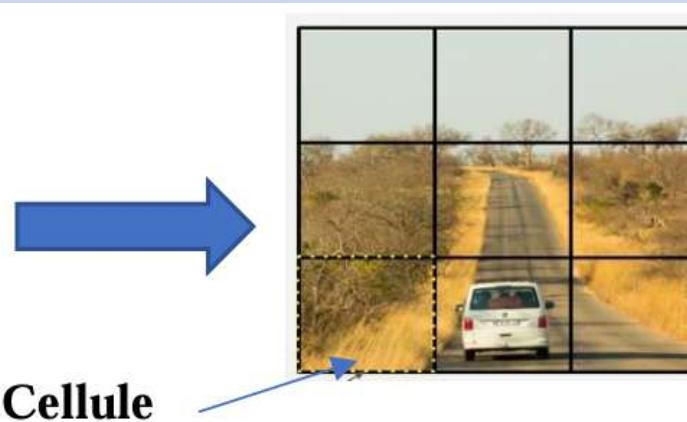
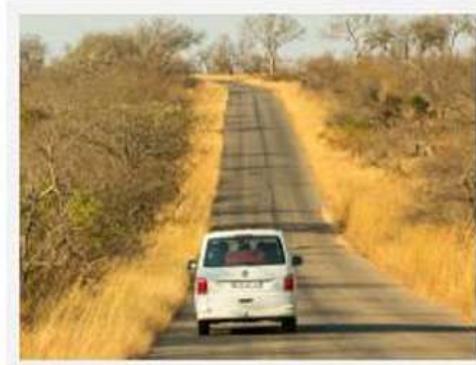


YOLO (You Only Look Once)

2. Diviser l'image

Ensuite, il divise l'image d'entrée en une grille de taille $S \times S$, comme un damier.

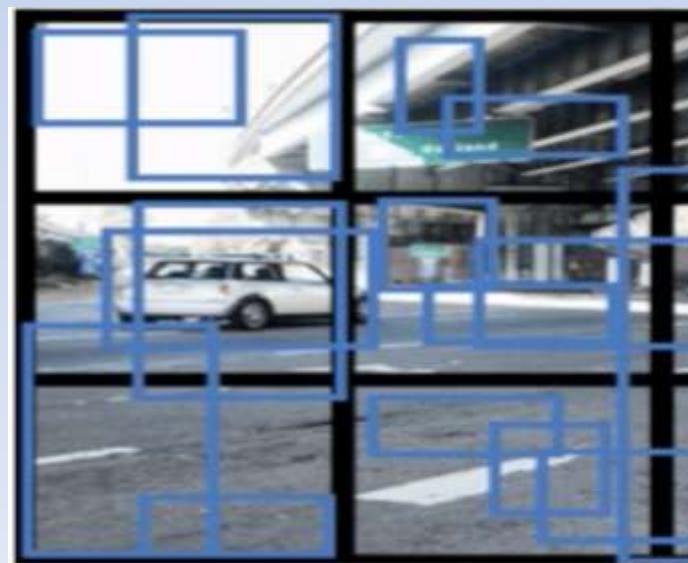
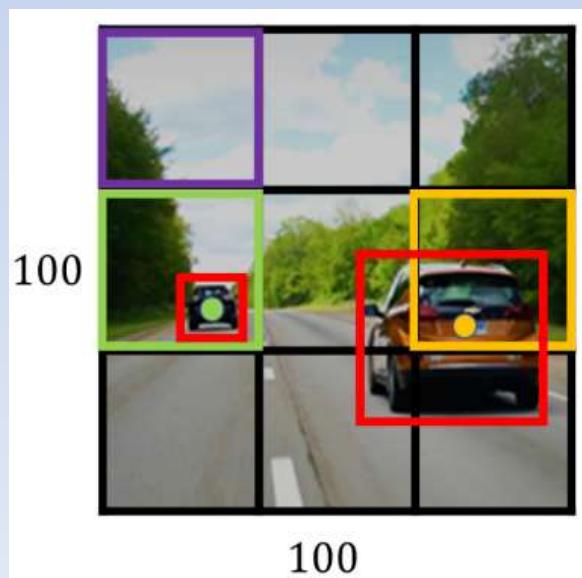
Chaque carré sera vérifié pour voir s'il contient un objet (un chat, un chien par exemple).



YOLO (You Only Look Once)

3. Chercher des indices:

Pour chaque cellule de la grille, YOLO recherche des indices ou des caractéristiques comme des contours, des formes ou des textures qui pourraient indiquer quel objet s'y trouve. Il les entoure avec des boîtes englobantes (**bounding boxes**).

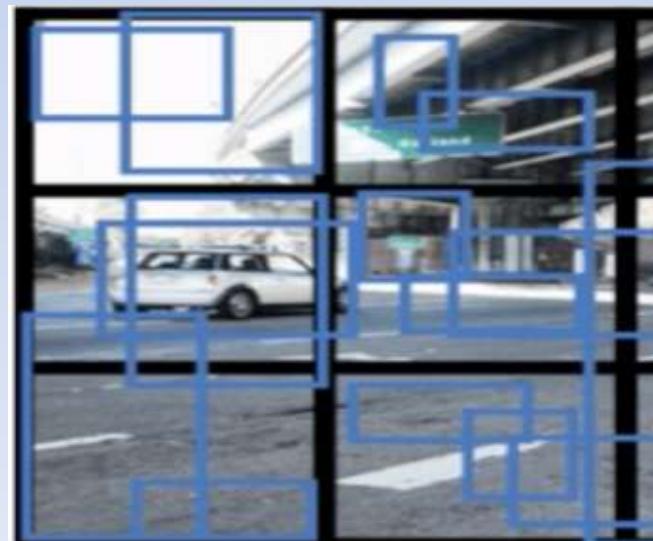
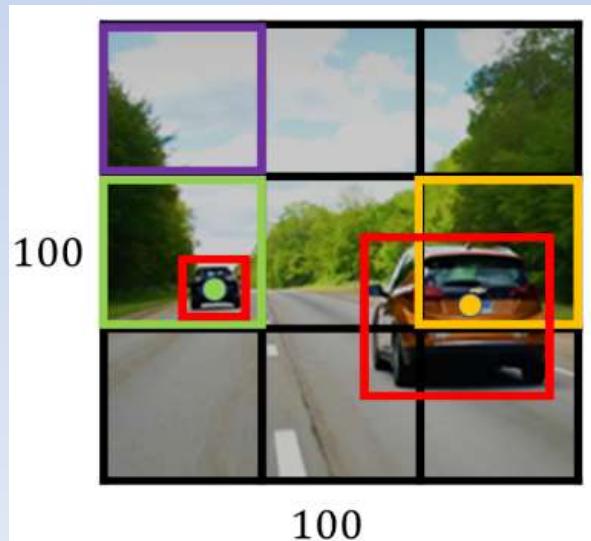


YOLO (You Only Look Once)

3. Chercher des indices

Pour chaque cellule de la grille, il prédit:

- **B** boites englobantes: **B=2** signifie qu'une cellule de la grille peut contenir au plus 2 objets.

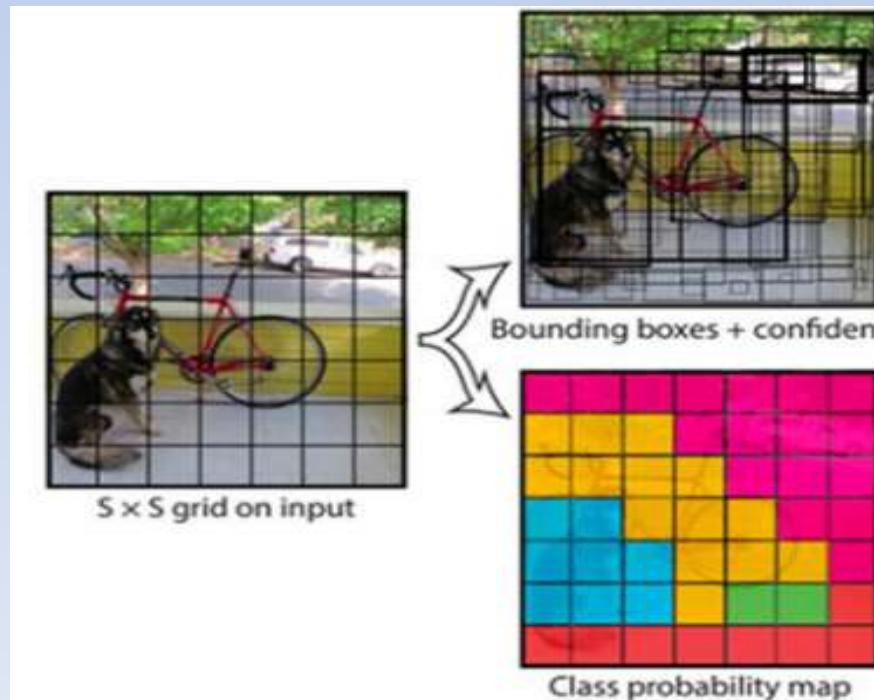


YOLO (You Only Look Once)

3. Chercher des indices

Pour chaque cellule de la grille, il prédit:

- Des scores de confiance pour ces boîtes.

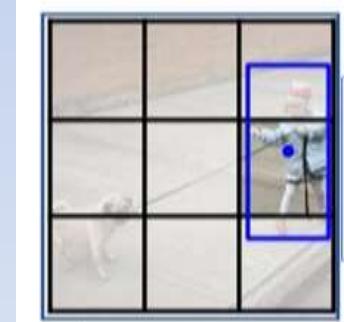


YOLO (You Only Look Once)

3. Chercher des indices

Chaque boite englobante se compose de 5 prédictions:

1. Quatre 4 coordonnées: x, y, w, h .
 - Les coordonnées (x, y) représentent le centre de la boîte par rapport aux limites de la cellule.
 - **La largeur et la hauteur** de la boite englobante.

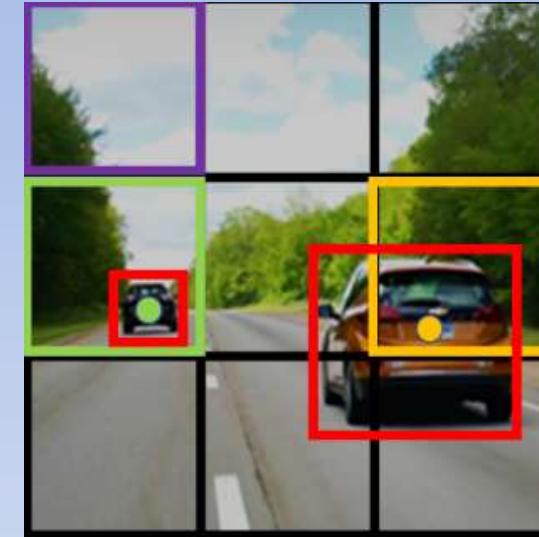


2. Un **score de confiance** pour la prédition associé aux différentes classes **C** choisies comme label.

YOLO (You Only Look Once)

Exemple:

- On considère une grille de taille 3x3.
- Un seul objet par cellule (donc $B=1$).
- On souhaite détecter 3 classes différentes (c_1, c_2 et c_3).
- Ici, la cellule violette ne nous intéresse pas, car elle ne possède aucun objet.



YOLO (You Only Look Once)

5. Éliminer les excédents

1. On supprime les Anchor boxes ayant une probabilité inférieure à un certain seuil, 0.6 par exemple.
2. On sélectionne l'anchor box avec la probabilité de détection la plus élevée ayant un **IoU** supérieur à un certain seuil, 0.5 généralement.

YOLO (You Only Look Once)

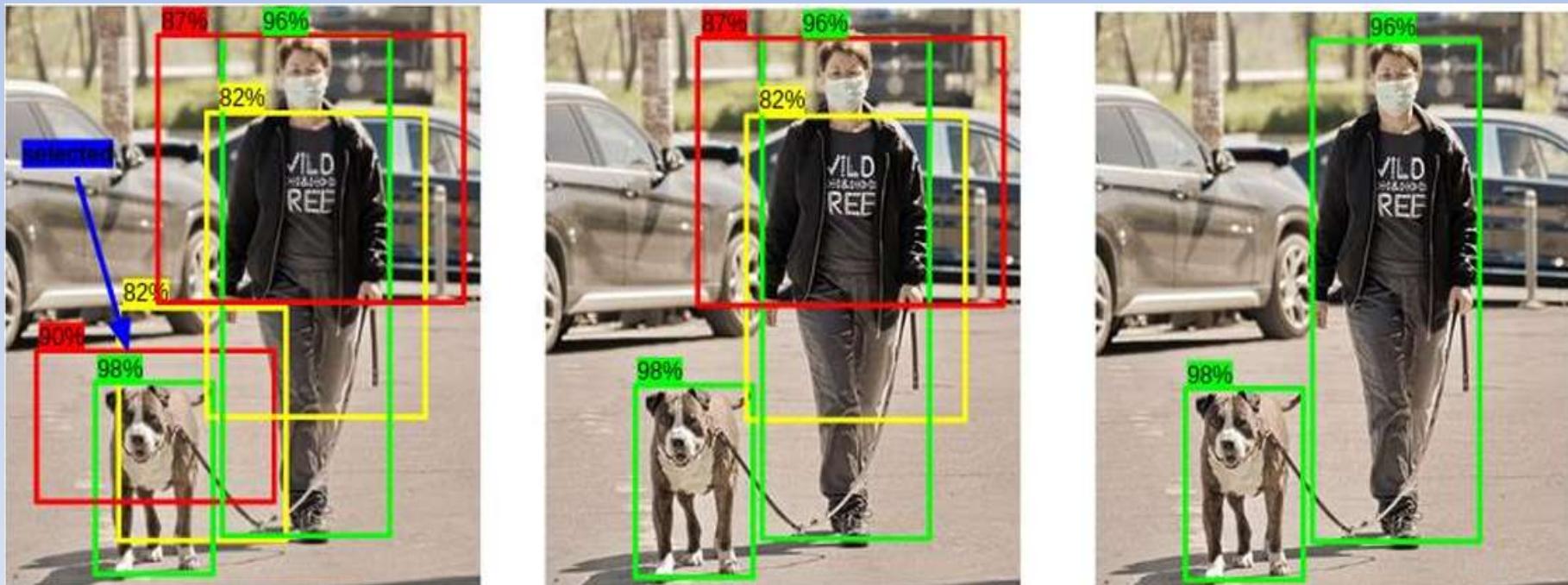
5. Éliminer les excédents

3. On supprime les anchor boxes qui intersectent l'anchor box sélectionnée en 2)

On supprime les anchor boxes trop proches les unes des autres dans cette étape car elles labelisent le même objet.

4. On repète 2) et 3).

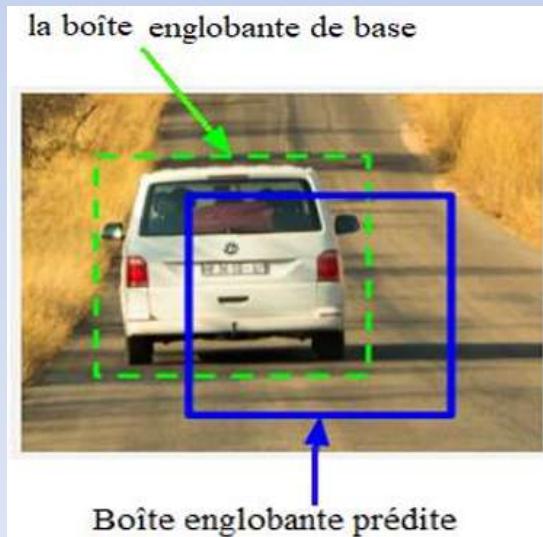
YOLO (You Only Look Once)



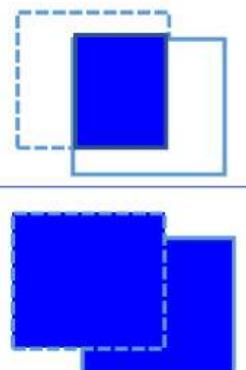
YOLO (You Only Look Once)

5. Éliminer les excédents

Dans l'étape 3, la suppression utilise un concept appelé « **Intersection over Union** » ou IoU. Il prend en entrée deux anchor boxes et il calcule le rapport de l'**intersection** et de l'**union** des deux.



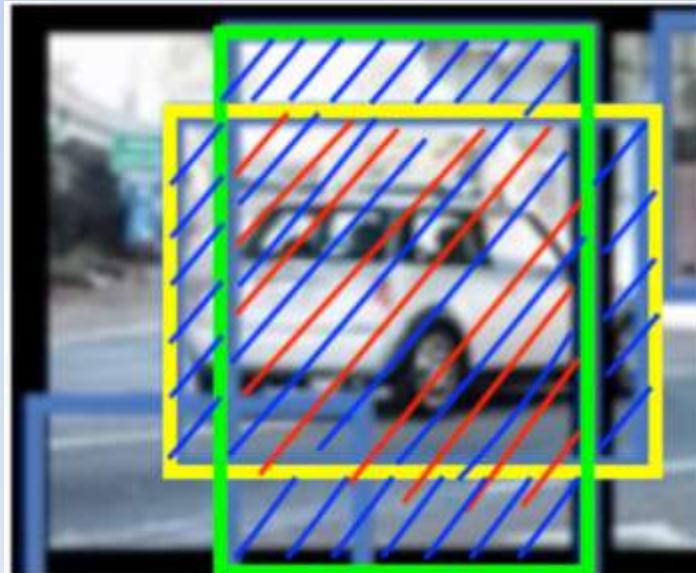
$$\text{Union sur Intersection} = \frac{\text{Zone de l'intersection}}{\text{Zone de l'Union}}$$



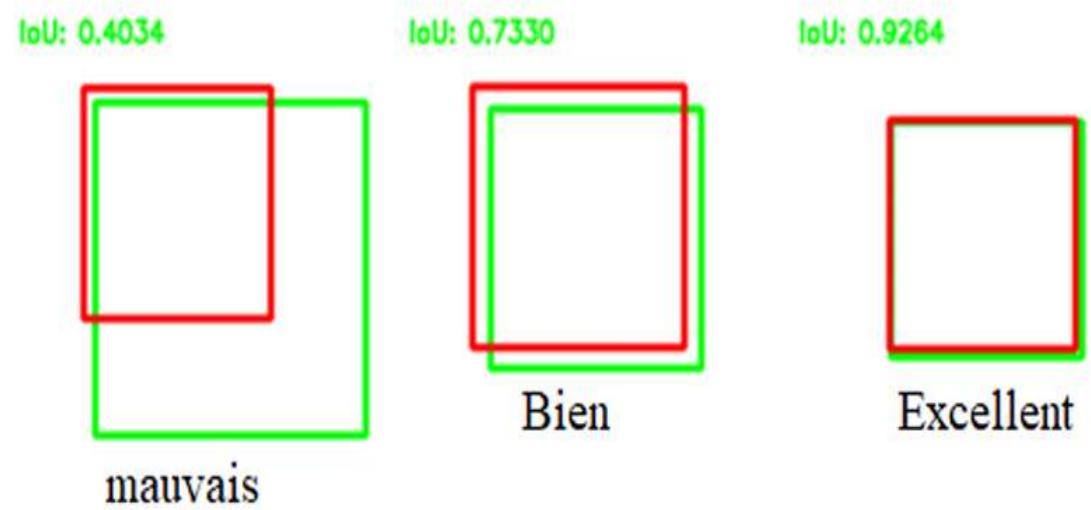
YOLO (You Only Look Once)

5. Éliminer les excédents

Dans l'étape 3, la suppression utilise un concept appelé « **Intersection over Union** » ou IoU. Il prend en entrée deux anchor boxes et il calcule le rapport de l'**intersection** et de l'**union** des deux.



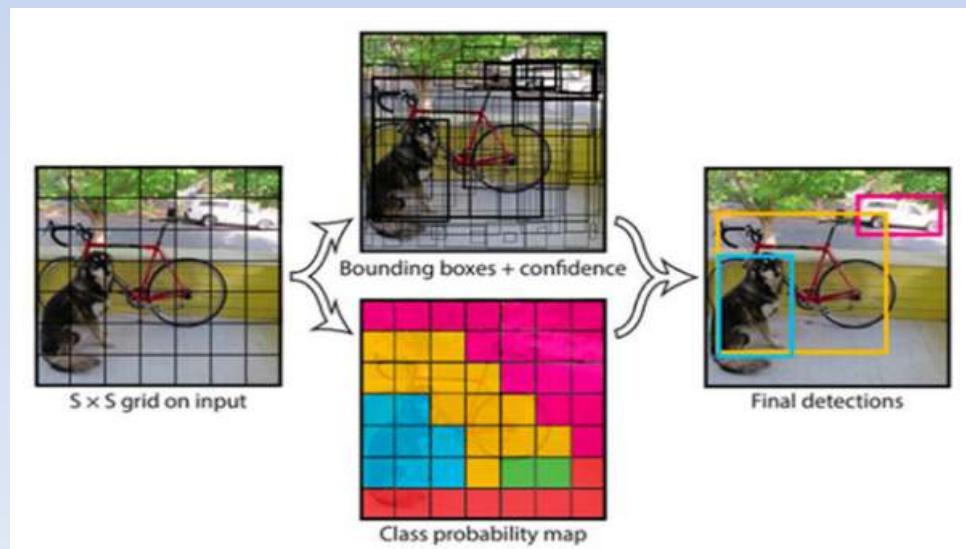
Union sur Intersection



YOLO (You Only Look Once)

6. Affichage du résultat

À la fin, YOLO vous montre où il pense que chaque objet se trouve en dessinant des boîtes autour d'eux et en les étiquetant, comme "voiture" ou "arbre".

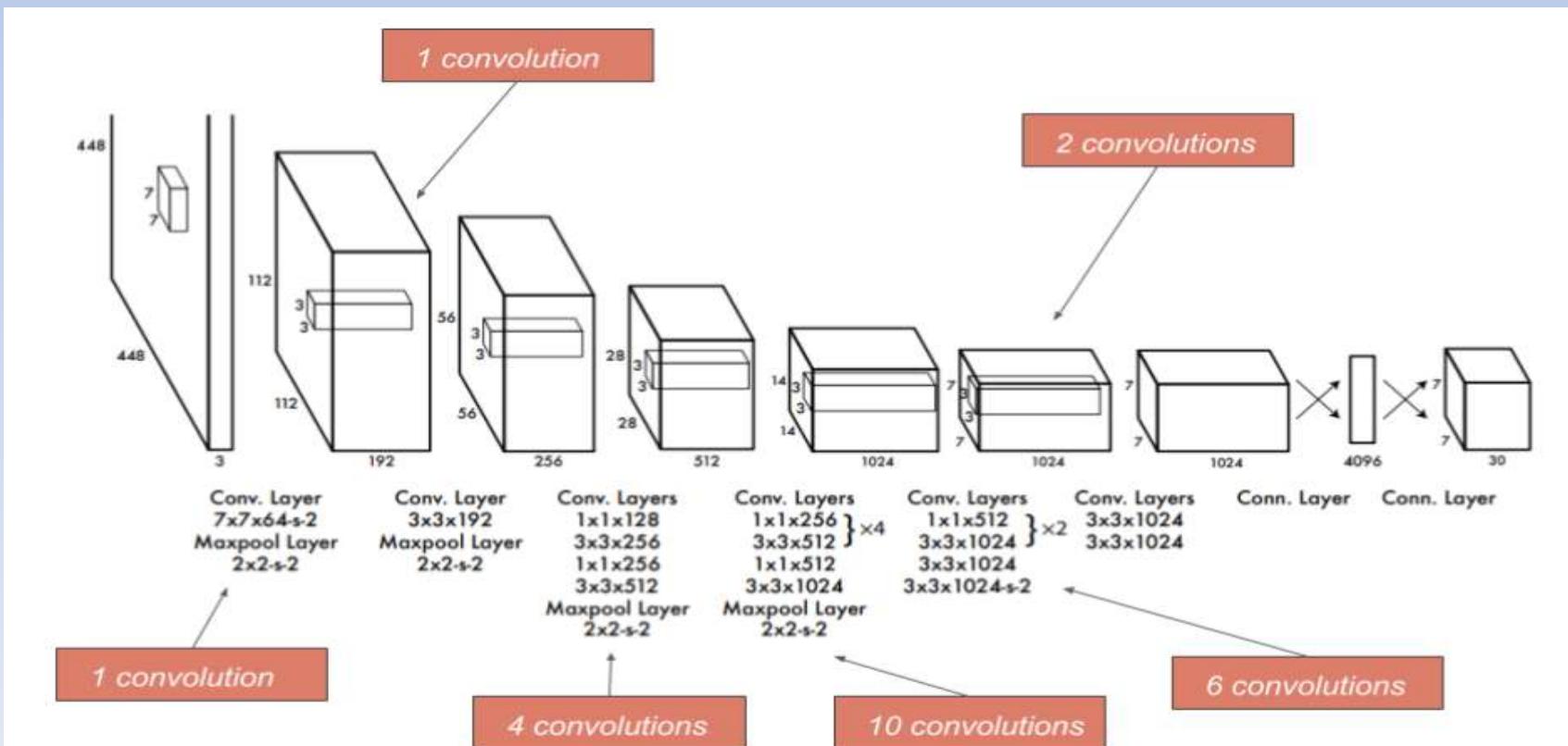


Architecture de YOLO

- L'architecture du modèle (du modèle GoogLeNet pour la classification d'images) se compose de:
24 couches convolutives pour extraire les features suivies de **2** couches denses entièrement connectées.
- L'entrée est une **image** de 448 x 448.
- La sortie est la **prédiction** de classe de l'objet enfermé dans la boîte englobante.

Architecture de YOLO

L'architecture du modèle se compose de: **24** couches convolutives pour extraire les features suivies de **2** couches denses entièrement connectées.



YOLO (You Only Look Once)

Le point fort de YOLO est:

- Il examine tous les éléments d'une images (décomposés en "carrés") en même temps.

C'est pourquoi il est rapide et peut même fonctionner en temps réel.

YOLO (You Only Look Once)

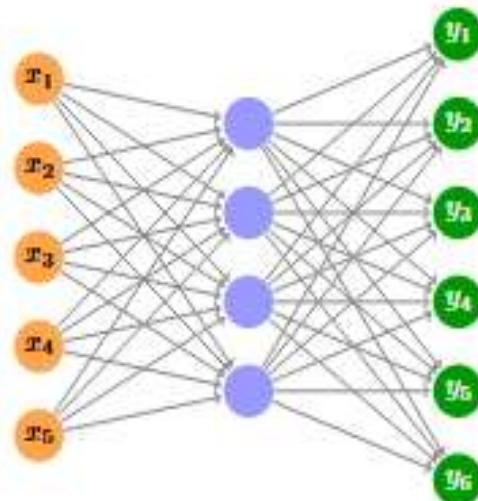
Les limitations du YOLO et ses variantes:

YOLO a du mal à détecter et à séparer **les petits objets** dans les images qui apparaissent en groupes, car chaque grille est contrainte de détecter un seul objet. Les petits objets qui viennent naturellement en groupes, comme une ligne de fourmis, sont donc difficiles à détecter et à localiser pour YOLO.

Réseau de neurone classique/
Réseau de neurone profond

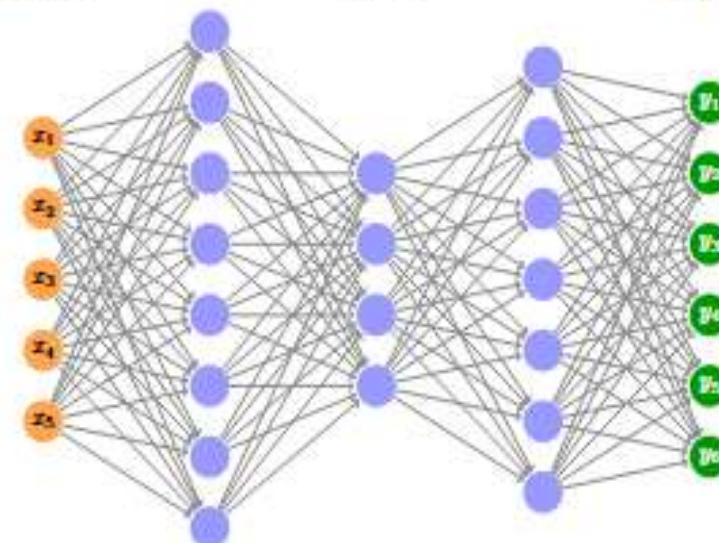
RN classique/ RN profond

Couche d'entrée Couche cachée Couche de sortie



(a) Réseau de neurones simple

Couche d'entrée Couche cachée Couche de sortie



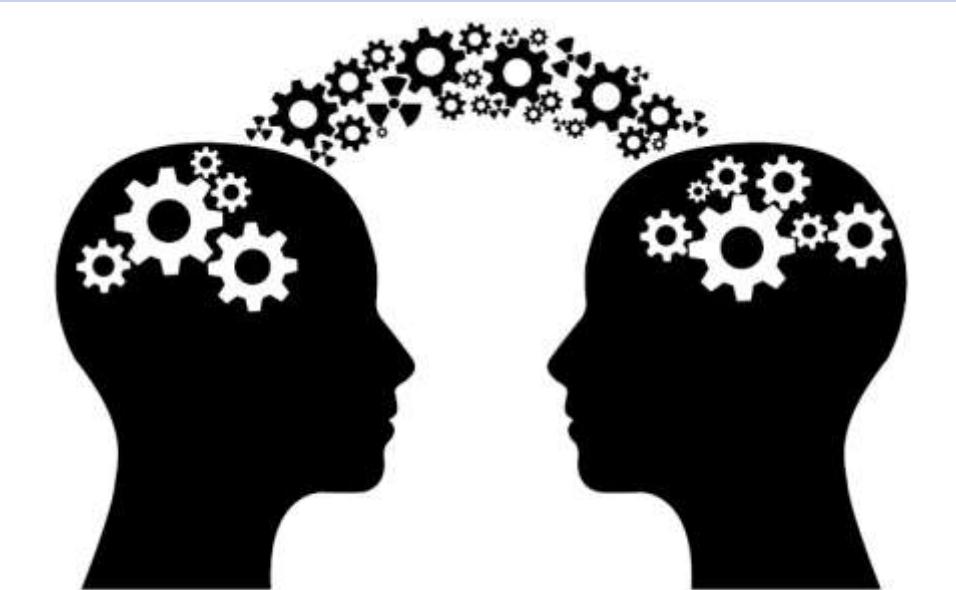
(b) Réseau de neurones profond

Quand utiliser les CNN?

- Si vous disposez d'une grande quantité de données complexes.
- Si vous voulez séparer l'extraction de caractéristiques de la classification.

Apprentissage par transfert

(transfer Learning)



Apprentissage par transfert

L'apprentissage par transfert est:

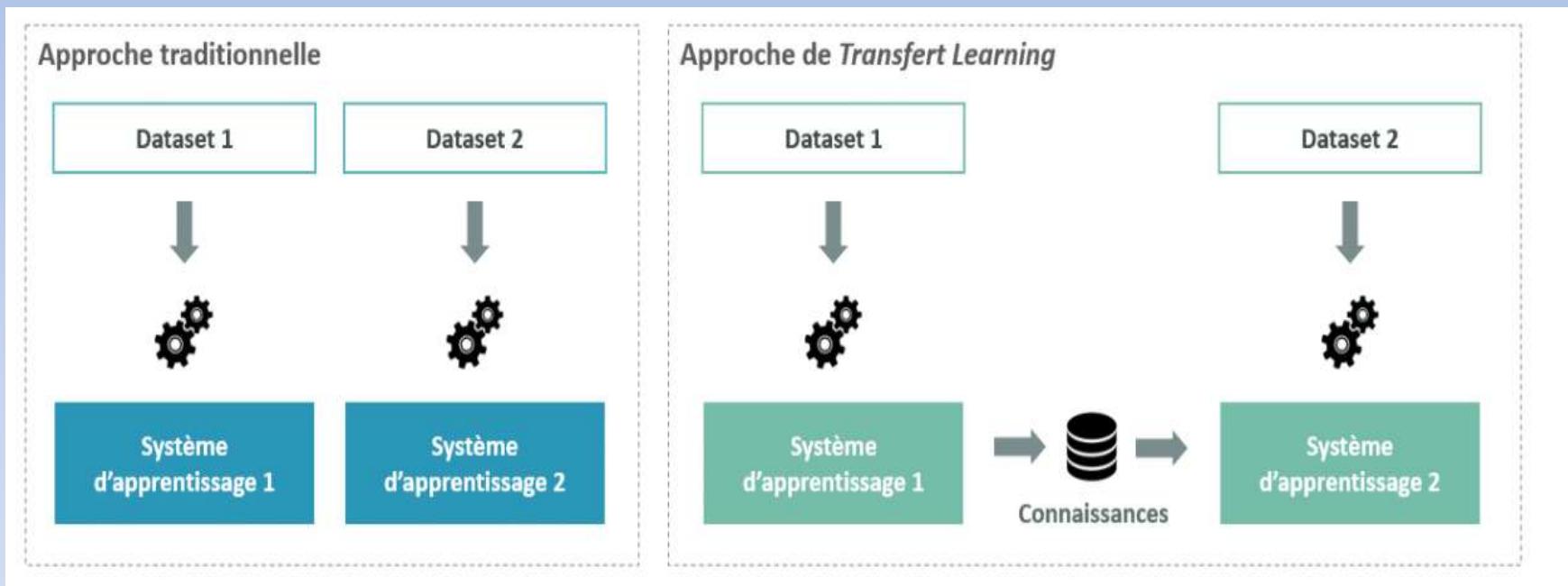
- Une technique d'apprentissage automatique.
- Les connaissances acquises à partir d'une tâche sont réutilisées pour améliorer les performances sur une mission similaire, mais plus précise.

Apprentissage par transfert

Il peut être vu comme:

- La capacité d'un système.
- A reconnaître et appliquer des connaissances et des compétences.
- Apprises à partir de tâches antérieures, sur de nouvelles tâches ou domaines partageant des similitudes.

Apprentissage par transfert



Apprentissage automatique traditionnel et apprentissage par transfert

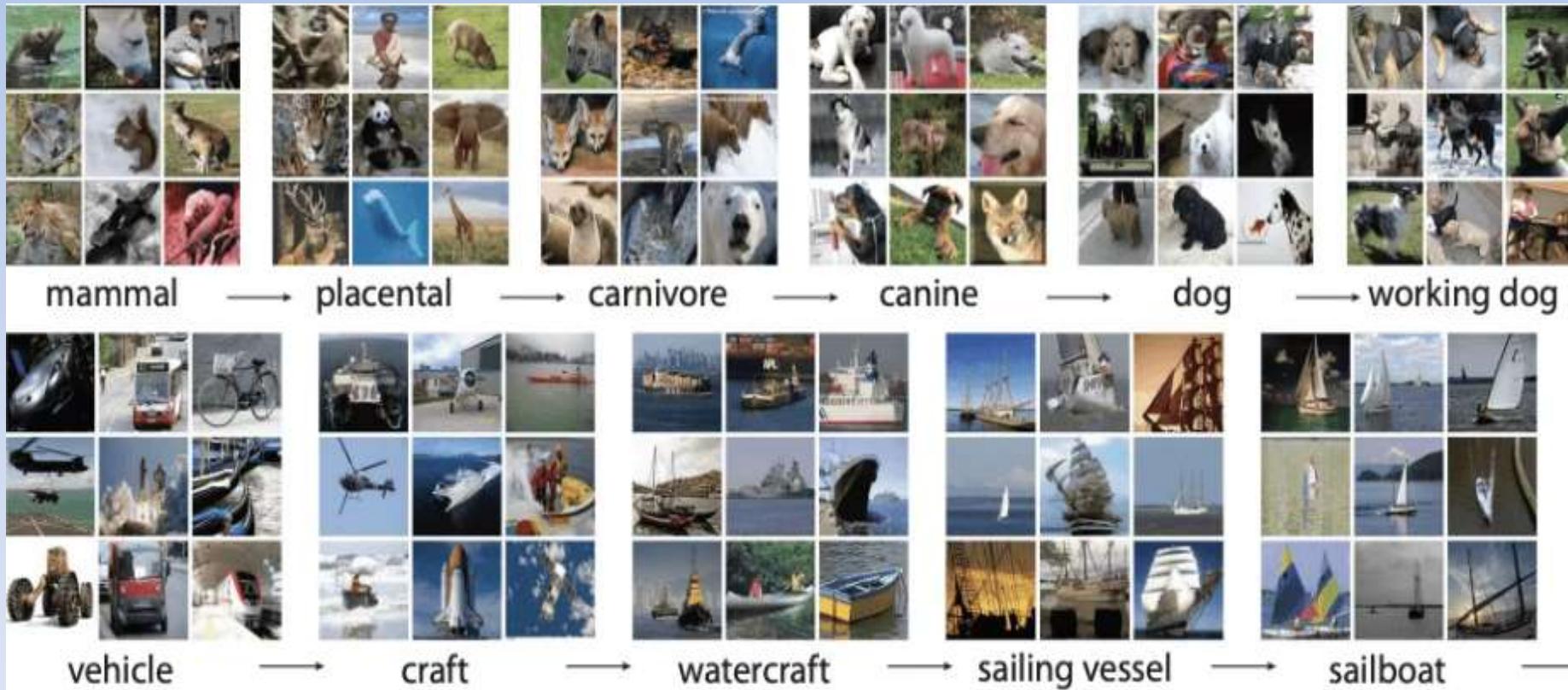
Apprentissage par transfert

Exemple:

- De nombreux CNN populaires sont pré-entraînés sur le jeu de données ImageNet.
- Il contient plus de **14 millions** d'images et **un millier de classes** d'images.

Apprentissage par transfert

Exemple:



Apprentissage par transfert

Exemple:

- Si vous devez classer les images des fleurs de votre jardin (ou toute autre image non incluse dans le jeu de données ImageNet).
- Vous disposez d'un nombre limité d'images de fleurs.

Apprentissage par transfert

Exemple:

- Vous pouvez transférer les couches et leurs pondérations. d'un réseau SqueezeNet.
- Remplacer les couches finales et ré-entraîner votre modèle avec les images dont vous disposez.

Apprentissage par transfert

Applications de l'apprentissage par transfert:

L'apprentissage par transfert est populaire dans de nombreuses applications de Deep Learning, telles que :

1. Computer Vision.



ResNet, une architecture de modèle préentraîné qui montre des performances améliorées dans les tâches de classification d'images et de détection d'objets.

Apprentissage par transfert

Applications de l'apprentissage par transfert:

L'apprentissage par transfert est populaire dans de nombreuses applications de Deep Learning, telles que :

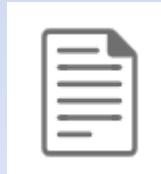
1. Computer Vision.



2. Le traitement de la parole et de l'audio.



3. Analyse de texte



Quels sont les avantages de l'AT ?

Efficacité améliorée

- La formation des modèles de machine learning prend du temps.
- Il nécessite également un grand ensemble de données et est coûteux en termes de calcul.

Quels sont les avantages de l'AT ?

Efficacité améliorée:

- Dans TL, un modèle pré-entraîné **conserve** les connaissances fondamentales des tâches, des caractéristiques, des poids et des fonctions, ce qui lui permet de **s'adapter** plus rapidement à de nouvelles tâches.
- Vous pouvez utiliser un ensemble de données beaucoup plus **restreint** et **moins de ressources** tout en obtenant de meilleurs résultats.

Quels sont les avantages de l'AT ?

Accessibilité accrue

La création de réseaux neuronaux d'apprentissage profond

nécessite

d'importants volumes de données, de ressources, de puissance de calcul et de temps.

Quels sont les avantages de l'AT ?

Accessibilité accrue

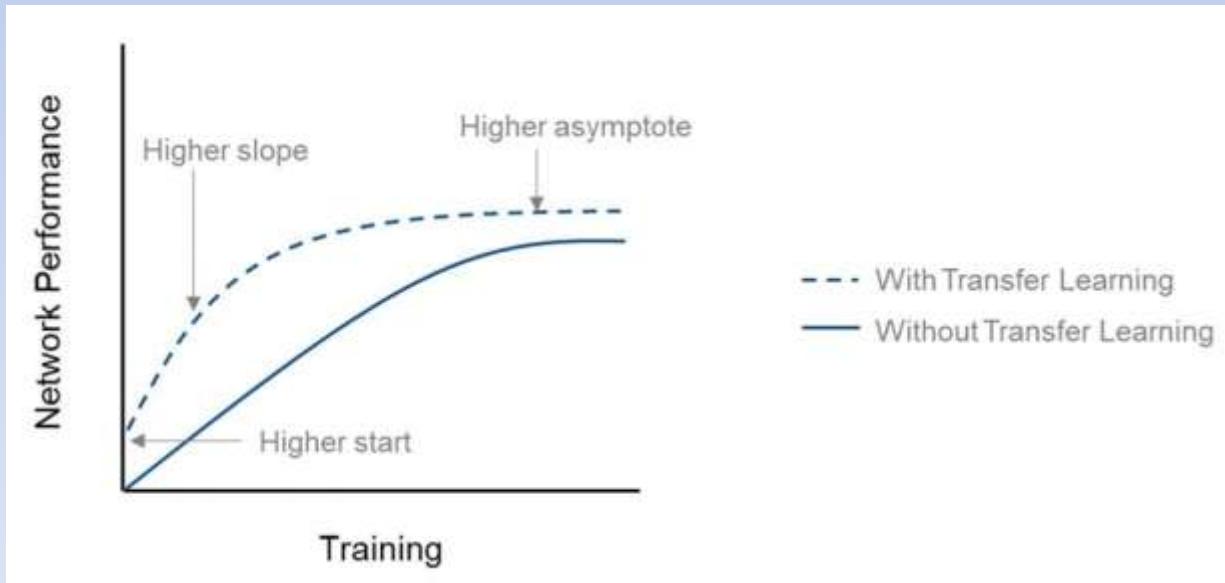
Vous pouvez adapter les modèles existants à vos besoins à une fraction du coût.

Par exemple, à l'aide d'un modèle de **reconnaissance d'image** préentraîné, vous pouvez créer des modèles pour l'analyse d'imagerie médicale, la surveillance environnementale ou la reconnaissance faciale avec un minimum d'ajustements.

Quels sont les avantages de l'AT ?

Performances améliorées:

Les modèles développés par le biais de TL font souvent preuve d'une plus grande robustesse dans des environnements divers et difficiles.



Comparaison de la précision de l'apprentissage à partir de zéro et de l'apprentissage par transfert.

Quels sont les avantages de l'AT ?

Performances améliorées

Les modèles développés par le biais de TL font souvent preuve d'une plus grande robustesse dans des environnements divers et difficiles.

Ils gèrent mieux la variabilité et le bruit du monde réel, ayant été exposés à un large éventail de scénarios lors de leur formation initiale.

Ils donnent de meilleurs résultats et s'adaptent de manière plus flexible aux conditions imprévisibles.¹³⁴

Quelles sont les conditions d'AT ?

L'apprentissage par transfert fonctionne mieux lorsque **trois** conditions sont réunies :

1. Les **deux tâches d'apprentissage** sont similaires.
2. La **distribution des données** dans les jeux de données source et cible a une variabilité faible.
3. Un modèle comparable peut être appliqué aux deux tâches.

Quelles sont les conditions d'AT ?

Si ces conditions ne sont pas remplies, l'apprentissage par transfert peut affecter négativement les performances du modèle.

On parle alors de *transfert négatif*.

Quelles sont les différentes stratégies d'AT ?

La stratégie que vous utiliserez pour faciliter l'AT dépendra du:

- Domaine du modèle que vous créez.
- La tâche à accomplir.
- La disponibilité des données d'entraînement.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert transductif

- Il implique le transfert de connaissances d'un domaine source spécifique vers un domaine cible différent mais connexe.
- L'accent étant mis principalement sur le domaine cible.
- Cela est particulièrement utile lorsque le domaine cible contient peu ou pas de données étiquetées.

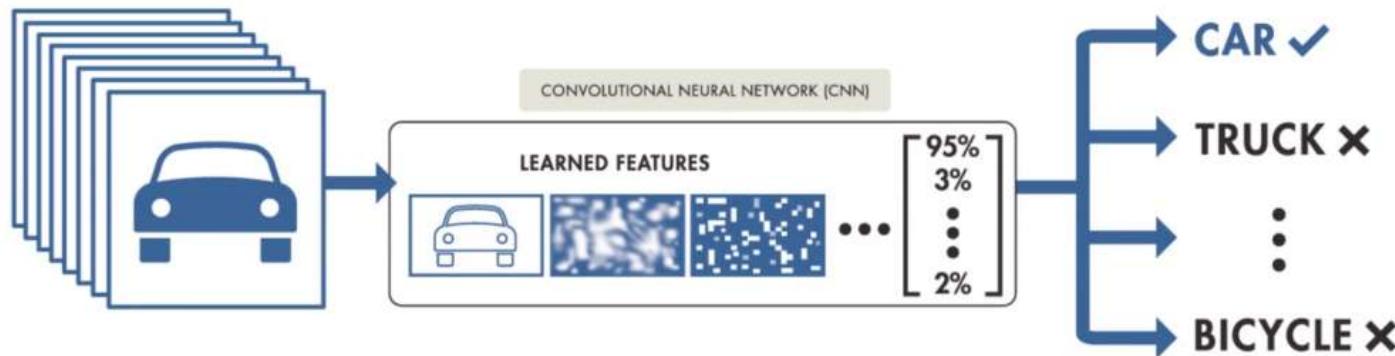
Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert transductif

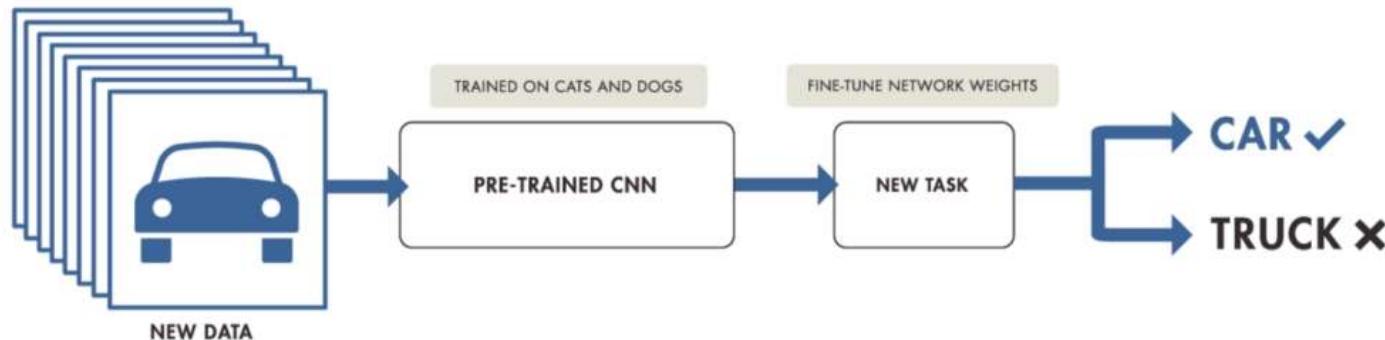
- Il demande au modèle de faire des prédictions sur des données cibles en utilisant les connaissances acquises précédemment.
- Les données cibles étant mathématiquement similaires aux données source, le modèle trouve des modèles et fonctionne plus rapidement.

Apprentissage par transfert

TRAINING FROM SCRATCH



TRANSFER LEARNING



Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert transductif

- Par exemple, envisagez d'adapter un modèle d'analyse des sentiments basé sur les critiques de produits pour analyser les critiques de films.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert transductif

- Le domaine source (**critiques de produits**) et le domaine cible (**critiques de films**) diffèrent en termes de contexte et de spécificités.
- Ils présentent des similitudes en termes de structure et d'utilisation de la langue.
- Le modèle apprend rapidement à appliquer sa compréhension du sentiment du domaine du produit au domaine du film.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert inductif

- Les domaines **source et cible** sont **identiques**, mais les tâches que le modèle doit accomplir sont différentes.
- **Exemple:** il est possible d'utiliser un modèle entraîné pour la détection d'animaux sur des images pour construire un modèle capable d'identifier des chiens.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert inductif

- Le traitement du langage naturel (NLP) est un exemple d'apprentissage par transfert inductif.
- Les modèles sont **préentraînés** sur un large ensemble de textes, puis **affinés** à l'aide de l'apprentissage par transfert inductif vers des fonctions spécifiques telles que l'analyse des sentiments.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert inductif

- De même, les modèles de vision par ordinateur tels que VGG sont préentraînés sur de grands ensembles de données d'images, puis affinés pour développer la détection d'objets.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert non supervisé

- Il utilise une stratégie similaire à l'apprentissage par transfert inductif.
- Il est utilisé lorsque vous ne disposez que de données non étiquetées dans les domaines source et cible.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert non supervisé

- Le modèle apprend les **caractéristiques communes** des données non étiquetées afin de les généraliser avec **plus de précision** lorsqu'on leur demande d'effectuer une tâche cible.
- Cette méthode est utile s'il est difficile ou coûteux d'obtenir des données sources étiquetées.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert non supervisé

- Par exemple, considérez la tâche consistant à identifier différents types de motos sur les images de la circulation.
- Dans un premier temps, le modèle est entraîné sur un grand nombre d'images de véhicules non étiquetées.

Quelles sont les différentes stratégies d'AT ?

Apprentissage par transfert non supervisé

Dans ce cas, le modèle détermine indépendamment les similitudes et les caractéristiques distinctives entre différents types de véhicules tels que les voitures, les bus et les motos.

- Ensuite, le modèle est présenté à un petit ensemble spécifique d'images de motos.
- Les performances du modèle s'améliorent considérablement par rapport à avant.

Apprentissage par transfert ou réglage fin

L'apprentissage par transfert se distingue du réglage fin.

- Tous deux réutilisent des modèles de machine learning préexistants au lieu de former de nouveaux modèles.
- Le **réglage fin** désigne le processus d'entraînement supplémentaire d'un modèle sur un jeu de données spécifique à une tâche, le but étant d'améliorer les performances de la tâche initiale pour laquelle le modèle a été construit.

Apprentissage par transfert ou réglage fin

Par exemple,

Il est possible de créer un modèle de détection d'objets à usage général en utilisant des ensembles d'images massifs tels que COCO ou ImageNet, puis d'entraîner le modèle résultant sur un jeu de données plus petit et étiqueté, spécifique à la détection de voitures.

Apprentissage par transfert ou réglage fin

Par exemple,

De cette manière, l'utilisateur effectue le réglage fin d'un modèle de détection d'objets, avec pour but la détection de voitures. En revanche, l'apprentissage par transfert signifie que les utilisateurs adaptent un modèle à un nouveau problème connexe plutôt qu'au même problème.

Quelles sont les étapes de l'AT ?

L'affinement d'un modèle d'apprentissage automatique pour une nouvelle tâche se déroule en trois étapes principales.

Sélectionner un modèle pré-entraîné



Configurer le modèle pré-entraîné



Entrainement du modèle cible

Quelles sont les étapes de l'AT ?

L'affinement d'un modèle d'apprentissage automatique pour une nouvelle tâche se déroule en trois étapes principales.

1. Sélectionner un modèle pré-entraîné:

Tout d'abord, sélectionnez un modèle préformé avec des connaissances ou des compétences préalables pour une tâche connexe.

Quelles sont les étapes de l'AT ?

1. Sélectionner un modèle pré-entraîné:

- Un contexte utile pour choisir un modèle approprié est de déterminer la tâche source de chaque modèle.
- Si vous comprenez les tâches initiales effectuées par le modèle, vous pouvez en trouver une qui permet de passer plus efficacement à une nouvelle tâche.

Quelles sont les étapes de l'AT ?

2. Configurer le modèle pré-entraîné:

- Après avoir sélectionné votre modèle source.
- Configurez-le pour transmettre les connaissances à un modèle afin d'effectuer la tâche correspondante.
- Il existe **deux méthodes** principales pour ce faire.

Quelles sont les étapes de l'AT ?

Il existe **deux méthodes** principales pour ce faire:

a. Congeler les couches préformées:

En **gelant les poids** (c'est-à-dire fixer les poids) des couches pré-entraînées, vous les maintenez fixes, préservant ainsi les connaissances que le modèle du deep learning a obtenues à partir de la tâche source.

Quelles sont les étapes de l'AT ?

b.1. Supprimer la dernière couche:

- Dans certains cas d'utilisation, vous pouvez également supprimer les dernières couches du modèle pré-entraîné.
- Dans la plupart des architectures ML, les dernières couches sont spécifiques aux tâches.
- La suppression de ces dernières couches vous permet de reconfigurer le modèle en fonction des nouvelles exigences des tâches.

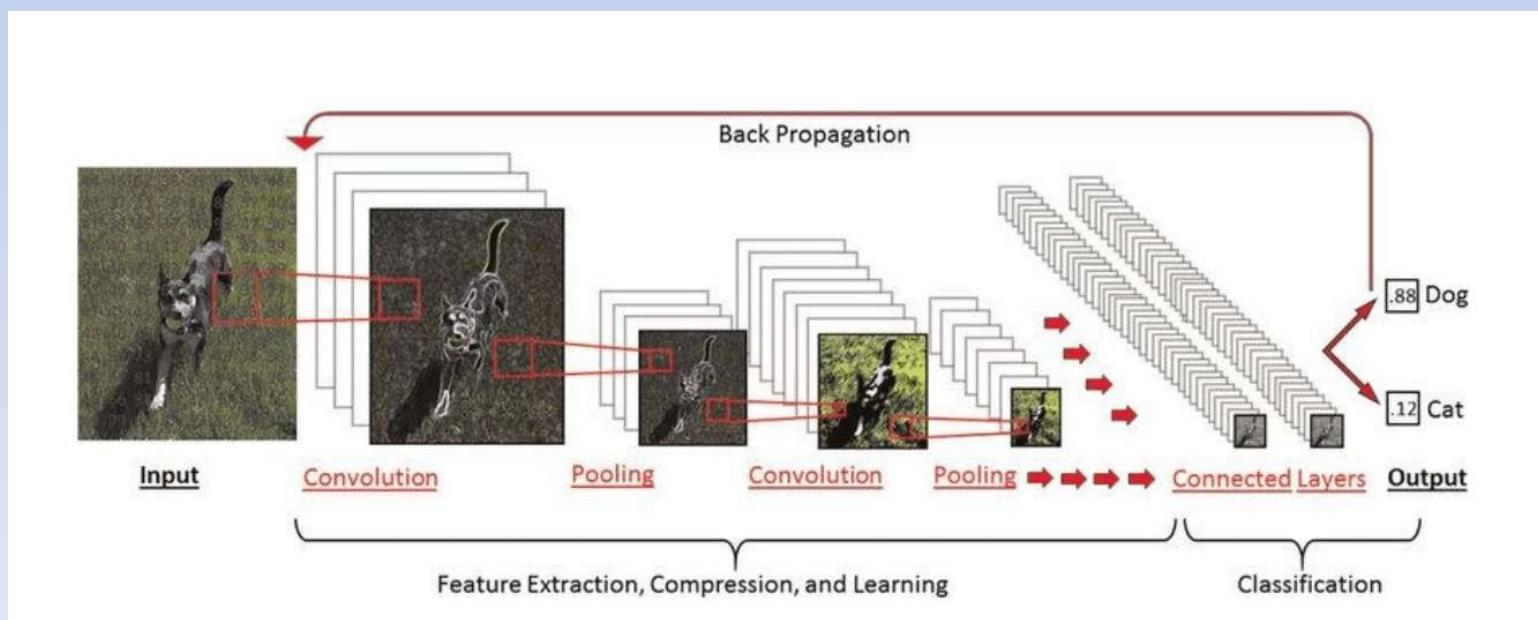
Quelles sont les étapes de l'AT ?

b.2. Introduire de nouvelles couches:

- L'ajout de nouvelles couches à votre modèle pré-entraîné vous aide à vous adapter à la nature spécialisée de la nouvelle tâche.
- Les nouvelles couches adaptent le modèle aux nuances et aux fonctions de la nouvelle exigence.

Quelles sont les étapes de l'AT ?

L'idée est donc de réutiliser un réseau pré-entraîné sans sa couche finale. Ce nouveau réseau fonctionne alors comme un extracteur de *features* fixes pour la réalisation d'autres tâches.



Exemple de modèle de réseau de neurones utilisé pour la classification d'images

Quelles sont les étapes de l'AT ?

3. Entraîner le modèle pour le domaine cible:

- Vous entraînez le modèle sur les données de tâches cibles afin de développer sa sortie standard afin de l'aligner sur la nouvelle tâche.
- Le modèle pré-entraîné produit probablement des résultats différents de ceux souhaités.

Quelles sont les étapes de l'AT ?

3. Entraîner le modèle pour le domaine cible

- Après avoir surveillé et évalué les performances du modèle pendant l'entraînement, vous pouvez ajuster les hyperparamètres ou l'architecture de base du réseau neuronal afin d'améliorer encore les résultats.

Quelles sont les étapes de l'AT ?

3. Entraîner le modèle pour le domaine cible

- Contrairement aux poids, les hyper-paramètres ne sont pas appris à partir des données.
- Ils sont prédéfinis et jouent un rôle crucial dans la détermination de l'efficience et de l'efficacité du processus de formation.

Quelles sont les étapes de l'AT ?

3. Entraîner le modèle pour le domaine cible

- Par exemple, vous pouvez ajuster les paramètres de régularisation ou les taux d'apprentissage du modèle pour améliorer ses capacités par rapport à la tâche cible.

Points clés à retenir

Le Transfer Learning correspond à:

La **capacité** à utiliser des connaissances **existantes**,
développées pour la résolution de problématiques données,
pour **résoudre** une nouvelle problématique.

Points clés à retenir

Il est important d'examiner les trois questions suivantes :

Que veut-on transférer ? C'est la première question à se poser et la plus importante.

Elle vise à définir quelle partie de la connaissance est à transférer de la source à la cible, afin de répondre à la problématique de la tâche cible.

Points clés à retenir

Il est important d'examiner les trois questions suivantes :

Quand veut-on transférer ? Il peut y avoir des scénarios où l'application de méthodes de Transfert Learning peut conduire à une dégradation des résultats.

En réalité, ces performances dépendent de la similarité entre les domaines et les tâches cible et source.

Points clés à retenir

Il est important d'examiner les trois questions suivantes :

Comment va-t-on réaliser le transfert ? Une fois que nous avons répondu à la question « quoi » et « quand », nous pouvons commencer à identifier la technique de Transfer Learning la plus adaptée au problème que nous souhaitons résoudre.

Merci pour votre attention