

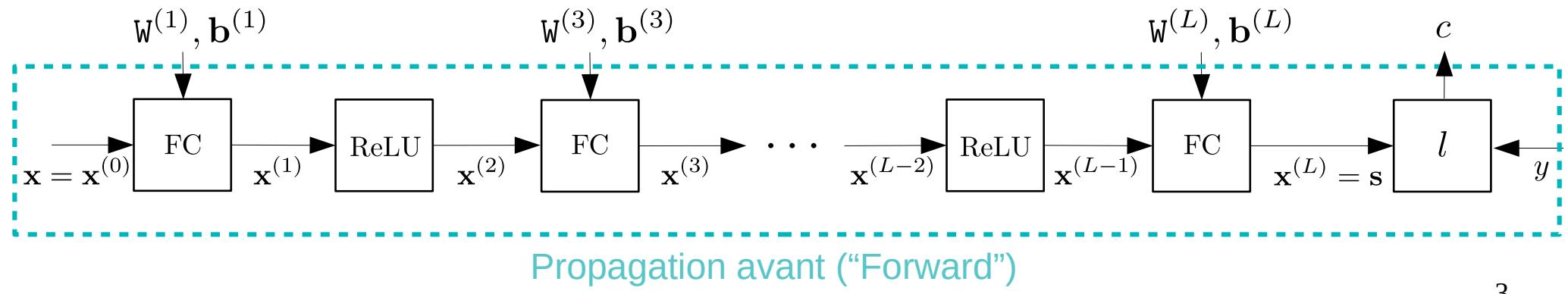
Réseaux de neurones à convolution

Guillaume Bourmaud

Rappel des ingrédients du « Deep Learning »

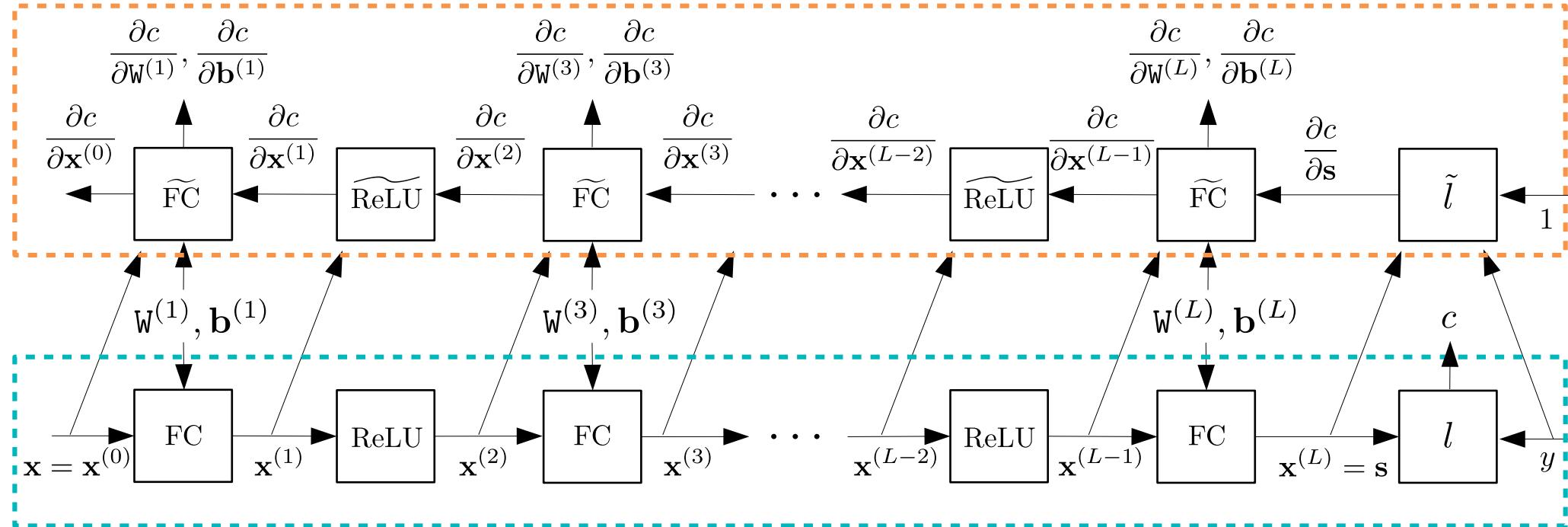
- 1) Grande base de données étiquetées
- 2) « Bonne » architecture de réseau de neurones profond
 - ▶ « Perceptron » multicouche, Réseau de neurones à convolution, Transformer
 - ▶ Et optimisation par descente de gradient stochastique (AdamW, etc.)
- 3) Grande capacité de calculs en parallèle (GPU)

Rappel MLP + Rétroprogation du gradient



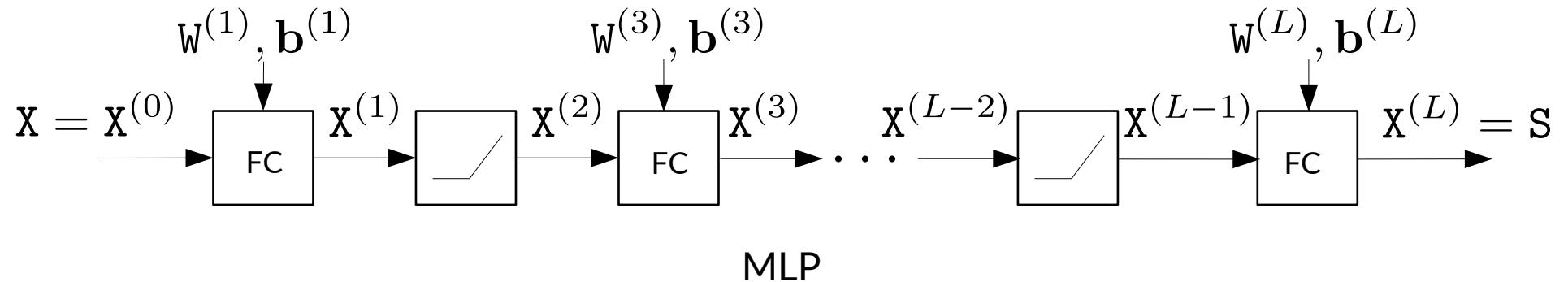
Rappel MLP + Rétroprogation du gradient

Rétropropagation (“Backward”)

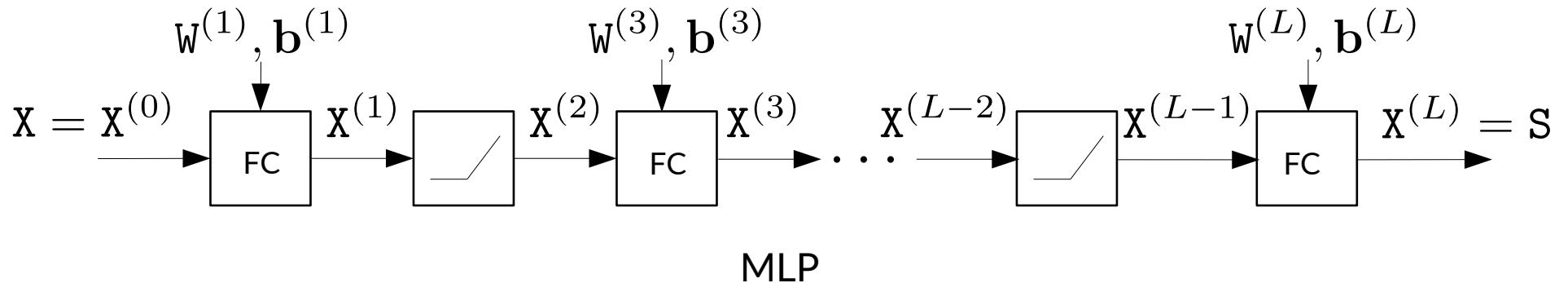


Propagation avant (Forward)

Introduction



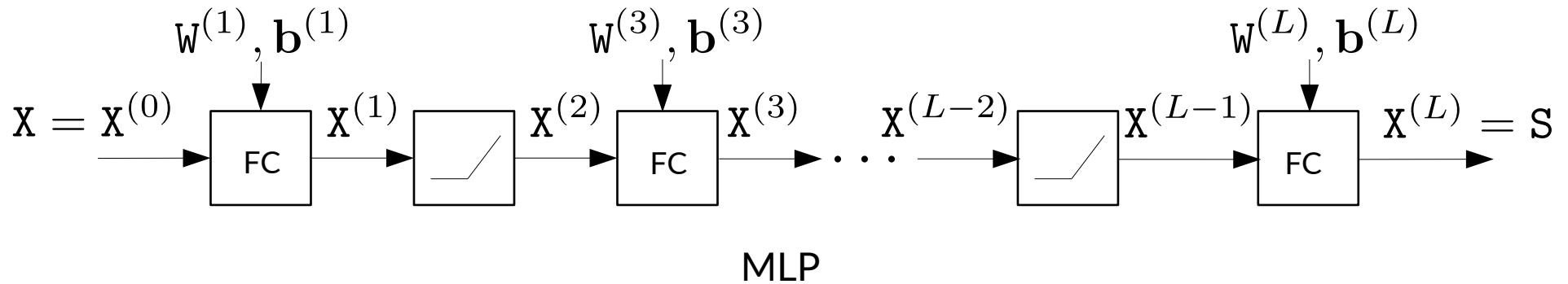
Introduction



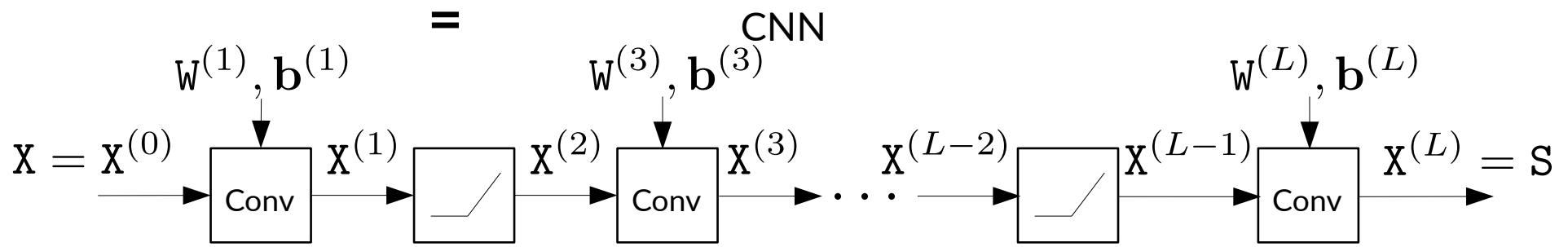
- FC (transformations affines générales)
- + Conv (transformations affines spécifiques)

= CNN

Introduction



- FC (transformations affines générales)
- + Conv (transformations affines spécifiques)



PLAN

I. Couche de convolution

II. Réseaux de neurones à convolution

I) Couche de convolution

I)

Limites d'une transformation affine générale (FC)

H=480



W=640

$\mathbf{x} : 640 \times 480 \times 3 \approx 10^6$ éléments

I)

Limites d'une transformation affine générale (FC)

$H=480$



$W=640$

$\mathbf{x} : 640 \times 480 \times 3 \approx 10^6$ éléments

Exemple d'une seule couche FC préservant la résolution de l'image d'entrée

$$\mathbf{y} = \mathbf{Wx} + \mathbf{b}$$

Diagram illustrating the computation of a single fully connected layer output \mathbf{y} from input \mathbf{x} . The input \mathbf{x} is a 3D volume of size $640 \times 480 \times 3$, represented as $10^6 \times 1$. The output \mathbf{y} is also a 3D volume of size $640 \times 480 \times 3$, represented as $10^6 \times 10^6$. The transformation is performed using a weight matrix \mathbf{W} of size $10^6 \times 10^6$ and a bias vector \mathbf{b} of size $10^6 \times 1$.

I)

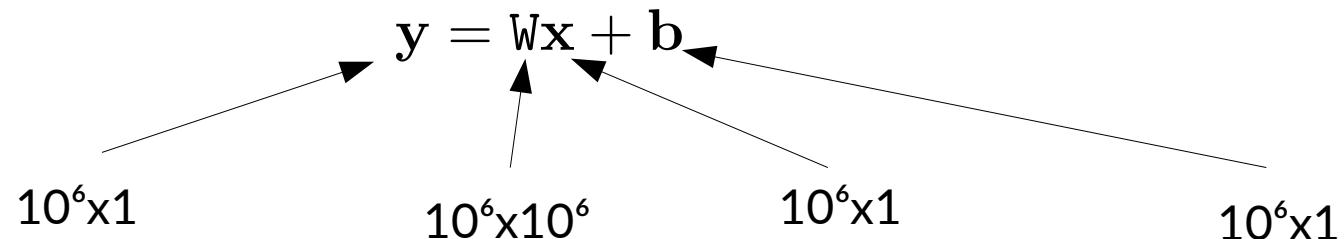
Limites d'une transformation affine générale (FC)



$\mathbf{x} : 640 \times 480 \times 3 \approx 10^6$ éléments

$W=640$

Exemple d'une seule couche FC préservant la résolution de l'image d'entrée



Occupation mémoire de \mathbf{W} : 4 octets (32 bits) $\times 10^6 \times 10^6 = 4\text{To}$.

Nombre de « multiplications+additions » également très élevé.

Opération de « convolution » = transformation affine spécifique

“Fully Connected”

=

Transformation affine générale

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} \\ W_{21} & W_{22} & W_{23} & W_{24} \\ W_{31} & W_{32} & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Opération de « convolution » = transformation affine spécifique

“Fully Connected”

=

Transformation affine générale

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} \\ W_{21} & W_{22} & W_{23} & W_{24} \\ W_{31} & W_{32} & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix}$$

Localement connecté à un
voisinage de taille 2

$$\begin{bmatrix} W_{11} & W_{12} & 0 & 0 \\ 0 & W_{22} & W_{23} & 0 \\ 0 & 0 & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix}$$

Opération de « convolution » = transformation affine spécifique

“Fully Connected”

=

Transformation affine générale

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} \\ W_{21} & W_{22} & W_{23} & W_{24} \\ W_{31} & W_{32} & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix}$$

Localement connecté à un voisinage de taille 2

$$\begin{bmatrix} W_{11} & W_{12} & 0 & 0 \\ 0 & W_{22} & W_{23} & 0 \\ 0 & 0 & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix}$$

« Convolution » avec filtre de taille 2

=

Équivariance par translation

$$\begin{bmatrix} W_{11} & W_{12} & 0 & 0 \\ 0 & W_{11} & W_{12} & 0 \\ 0 & 0 & W_{11} & W_{12} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix}$$

Opération de « convolution » = transformation affine spécifique

“Fully Connected”

=

Transformation affine générale

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} \\ W_{21} & W_{22} & W_{23} & W_{24} \\ W_{31} & W_{32} & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Localement connecté à un voisinage de taille 2

$$\begin{bmatrix} W_{11} & W_{12} & 0 & 0 \\ 0 & W_{22} & W_{23} & 0 \\ 0 & 0 & W_{33} & W_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

« Convolution » avec filtre de taille 2
=
Équivariance par translation

$$\begin{bmatrix} W_{11} & W_{12} & 0 & 0 \\ 0 & W_{11} & W_{12} & 0 \\ 0 & 0 & W_{11} & W_{12} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix}$$

Bonne idée pour tous les types de signaux (son, image, etc.), il faut que la notion de voisinage ait un sens !

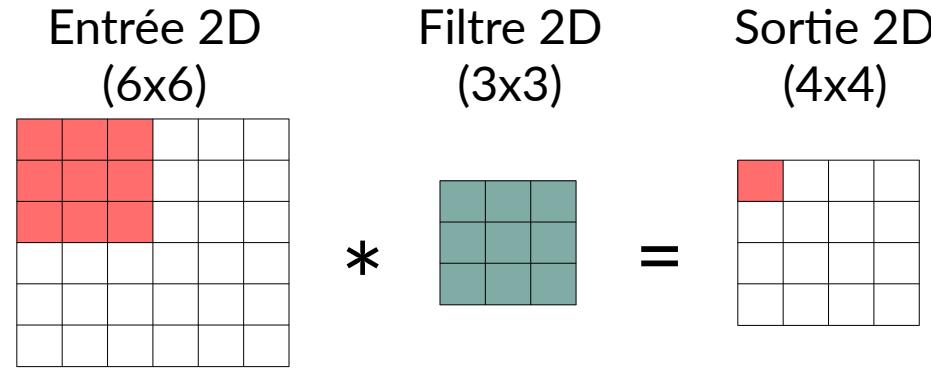
→ Beaucoup moins de paramètres à stocker

→ Beaucoup moins de « multiplications+additions »

I)

Opération de « convolution » en 2D

En fait, il s'agit d'une
intercorrélation

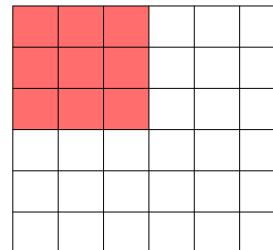


I)

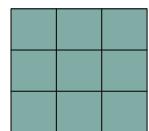
Opération de « convolution » en 2D

En fait, il s'agit d'une
intercorrélation

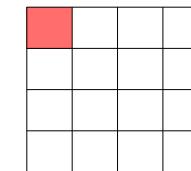
Entrée 2D
(6x6)



Filtre 2D
(3x3)

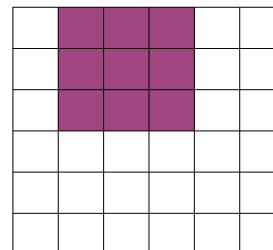


Sortie 2D
(4x4)



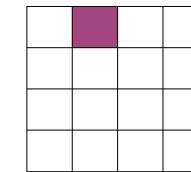
*

=



*

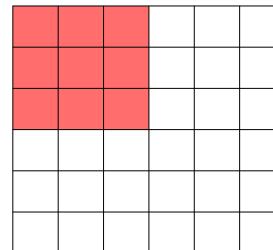
=



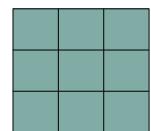
Opération de « convolution » en 2D

En fait, il s'agit d'une
intercorrélation

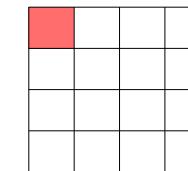
Entrée 2D
(6x6)



Filtre 2D
(3x3)

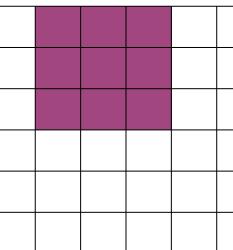


Sortie 2D
(4x4)



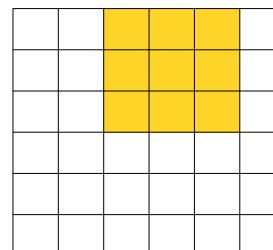
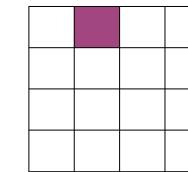
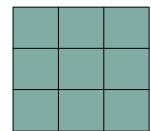
*

=



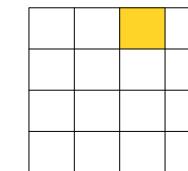
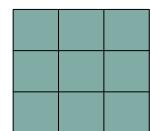
*

=



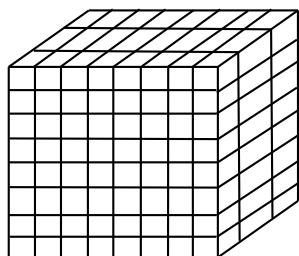
*

=



I)

Couche de convolution 2D : un seul filtre



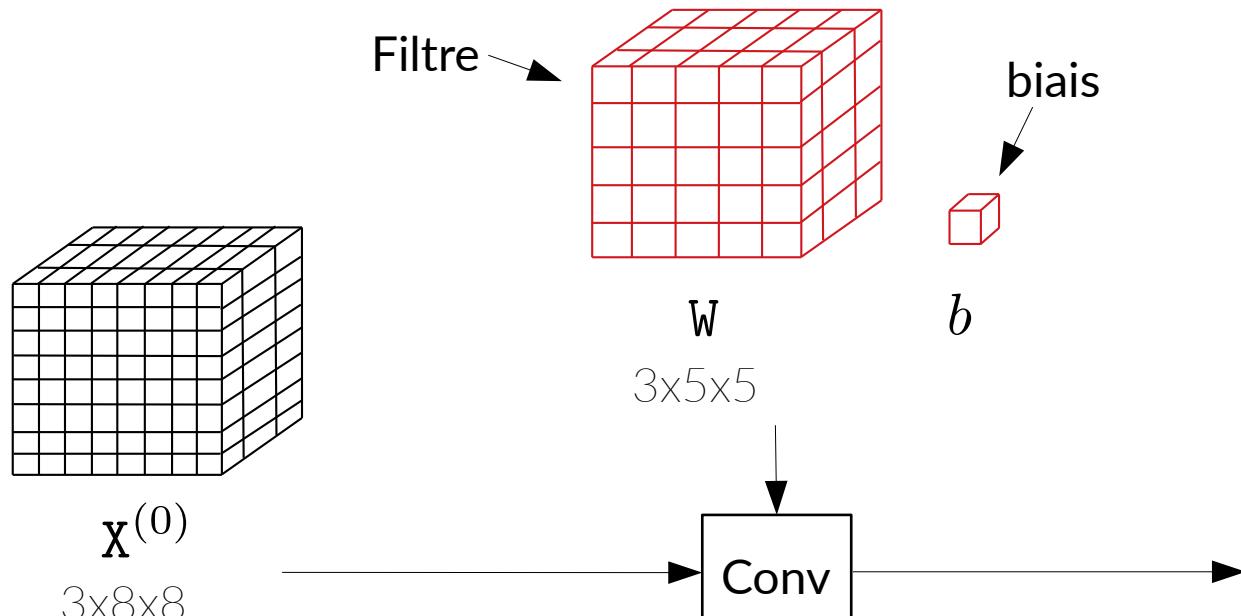
$x^{(0)}$

3x8x8

« Tenseur »
d'entrée

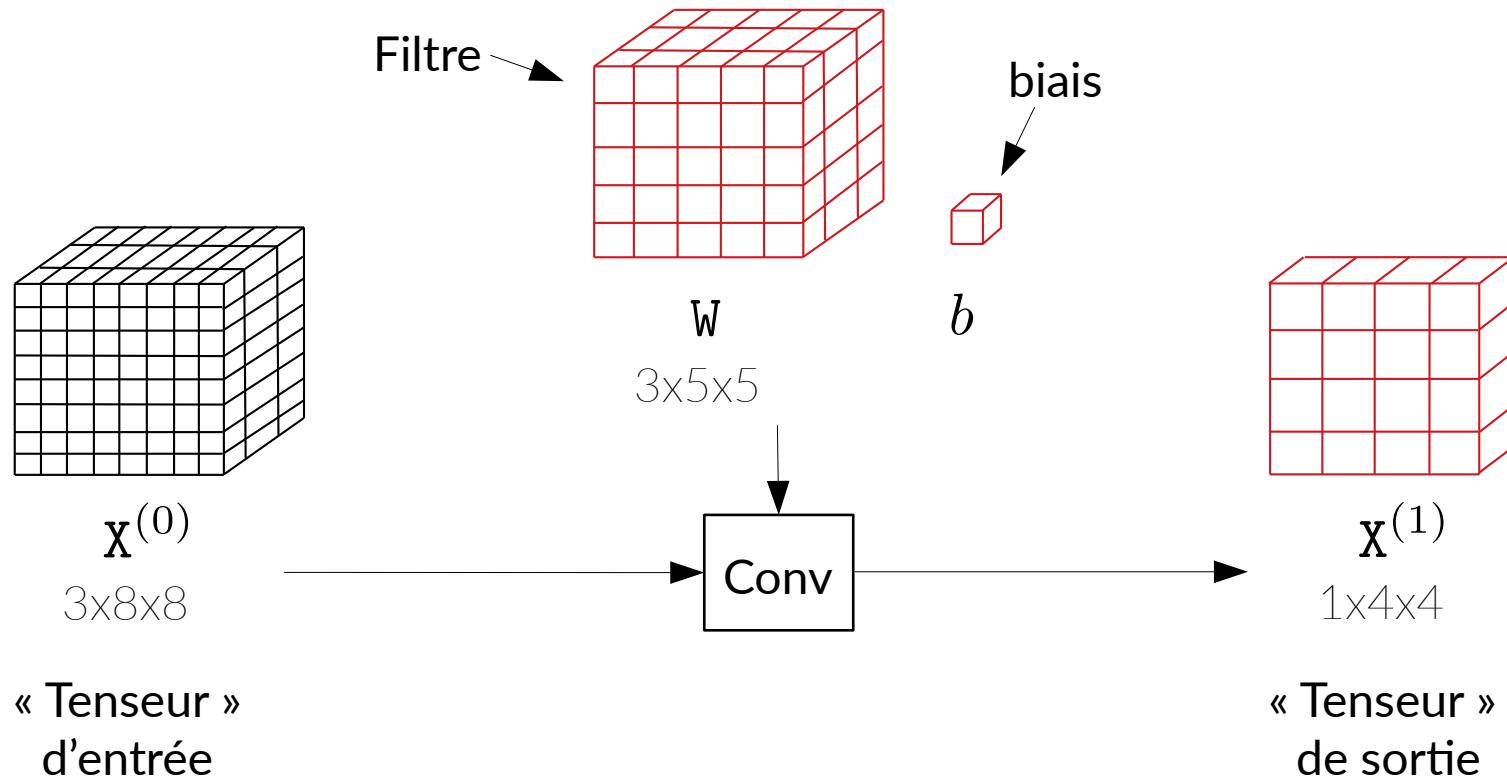
« Tenseur » = tableau multi-dimensionnel

Couche de convolution 2D : un seul filtre

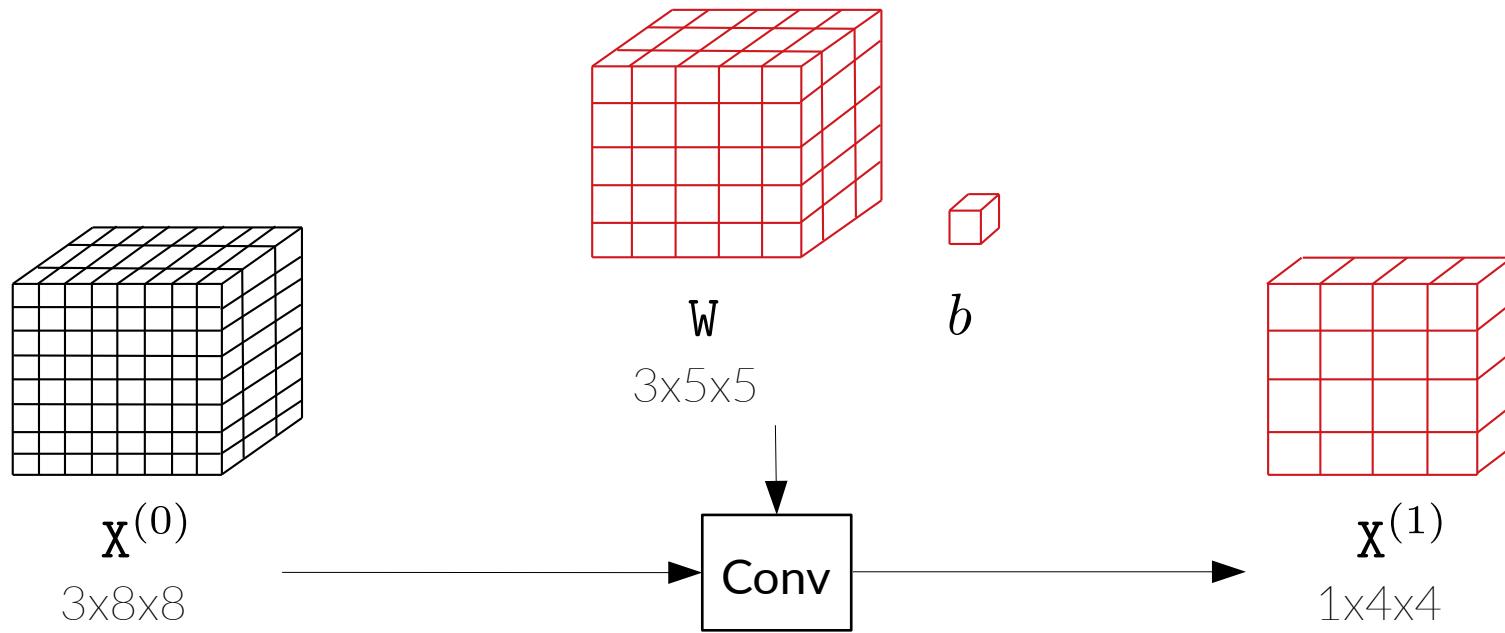


« Tenseur »
d'entrée

Couche de convolution 2D : un seul filtre

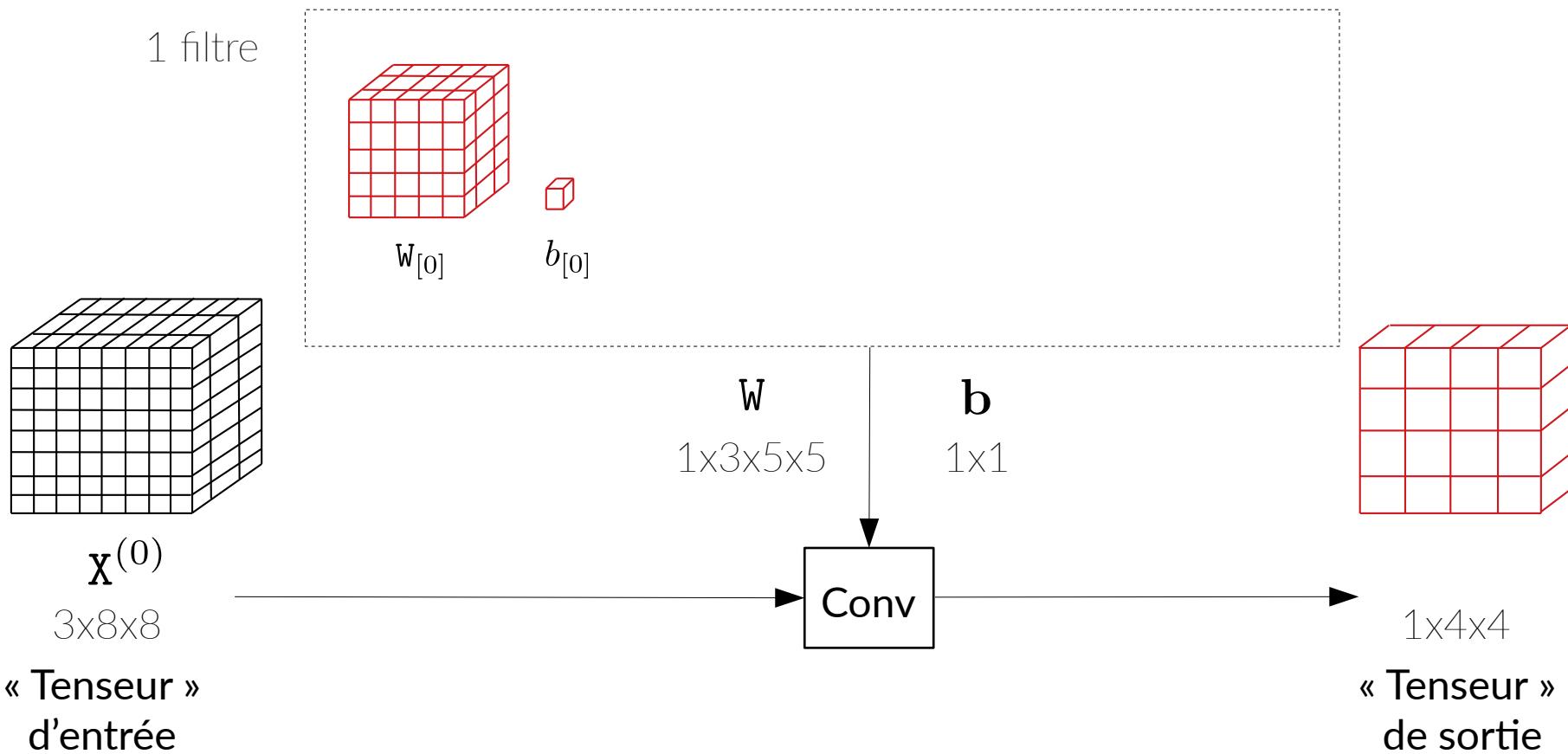


Couche de convolution 2D : un seul filtre

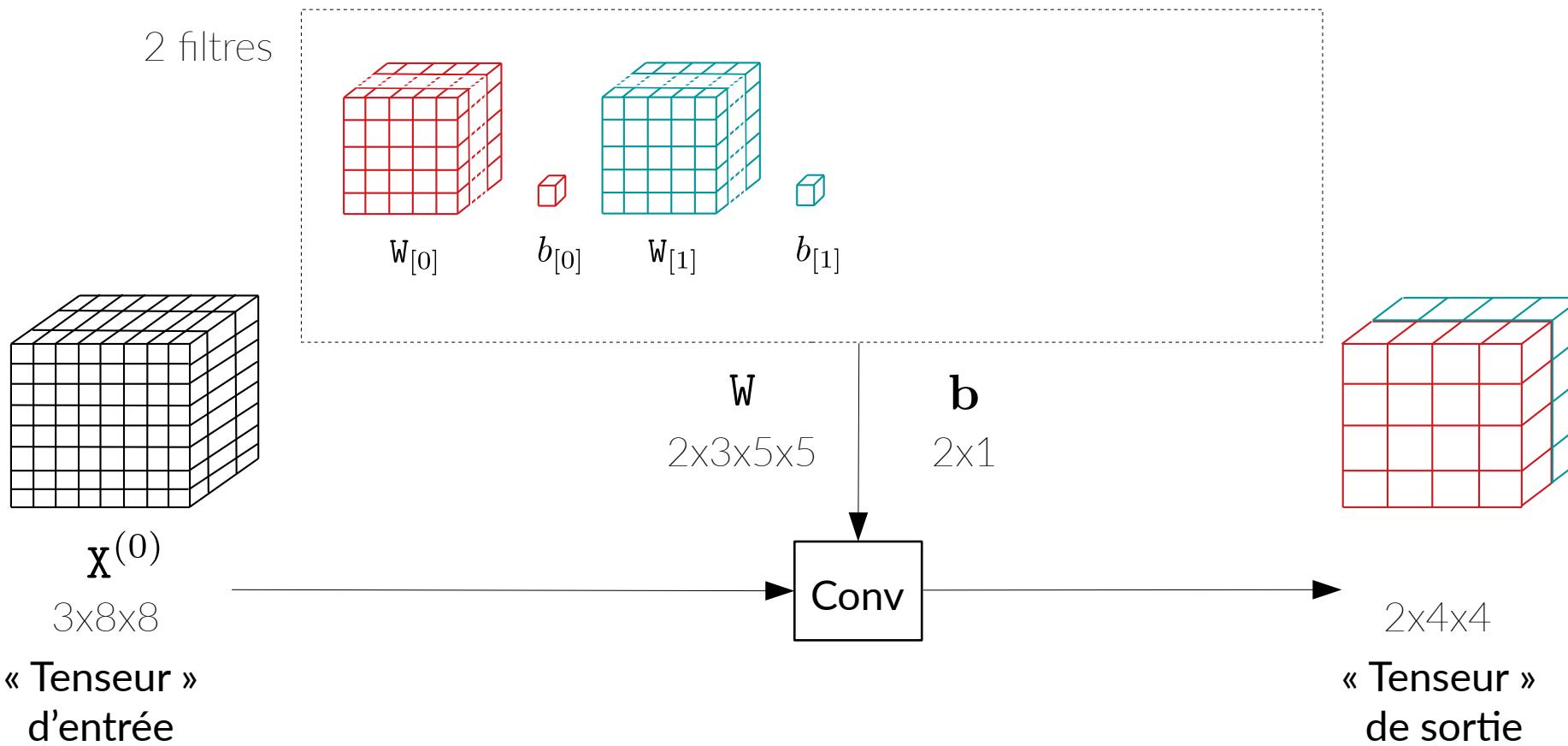


$$X_{i,j}^{(1)} = \sum_{k=0}^2 \sum_{m=0}^4 \sum_{n=0}^4 w_{k,m,n} X_{k,i+m,j+n}^{(0)} + b$$

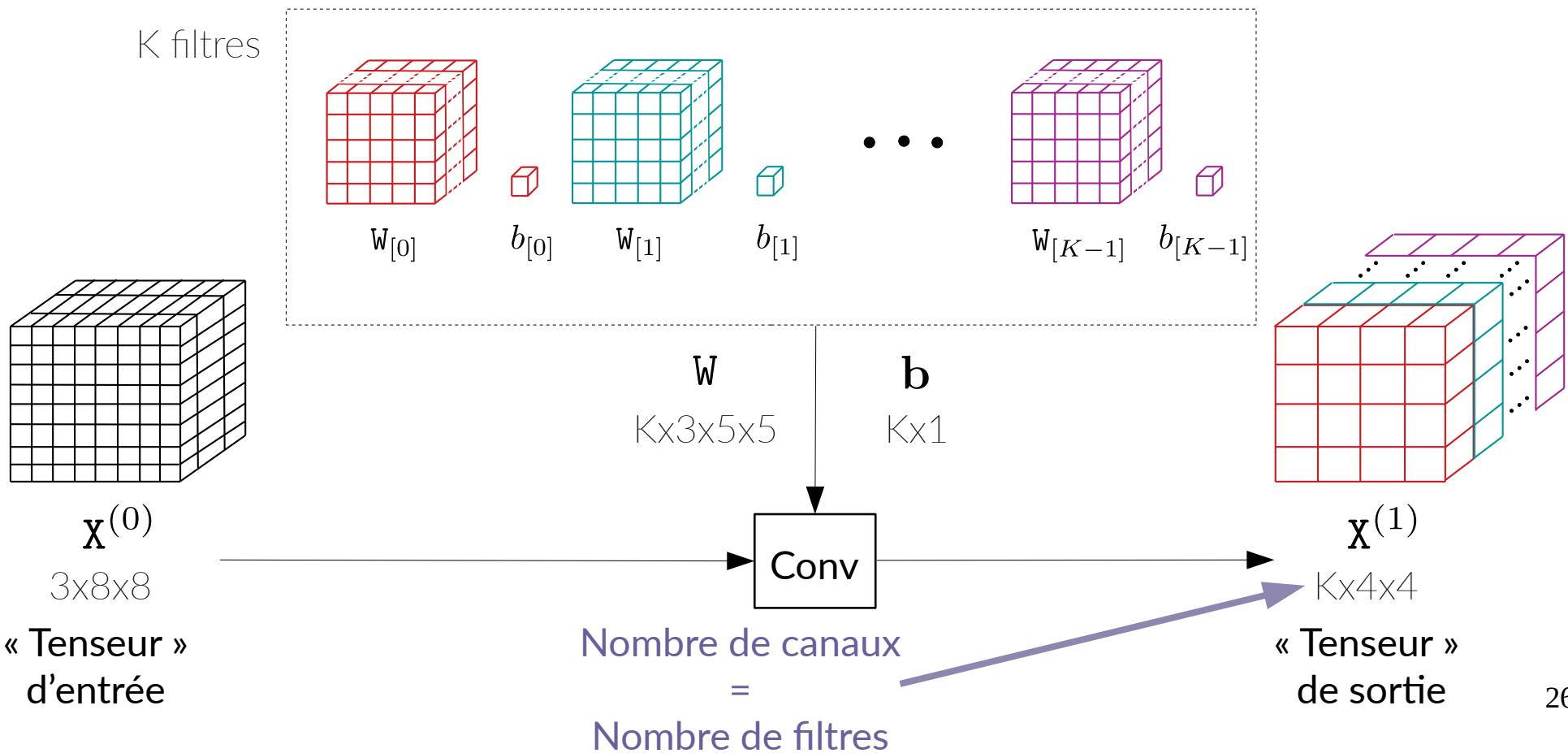
Couche de convolution 2D : K filtres



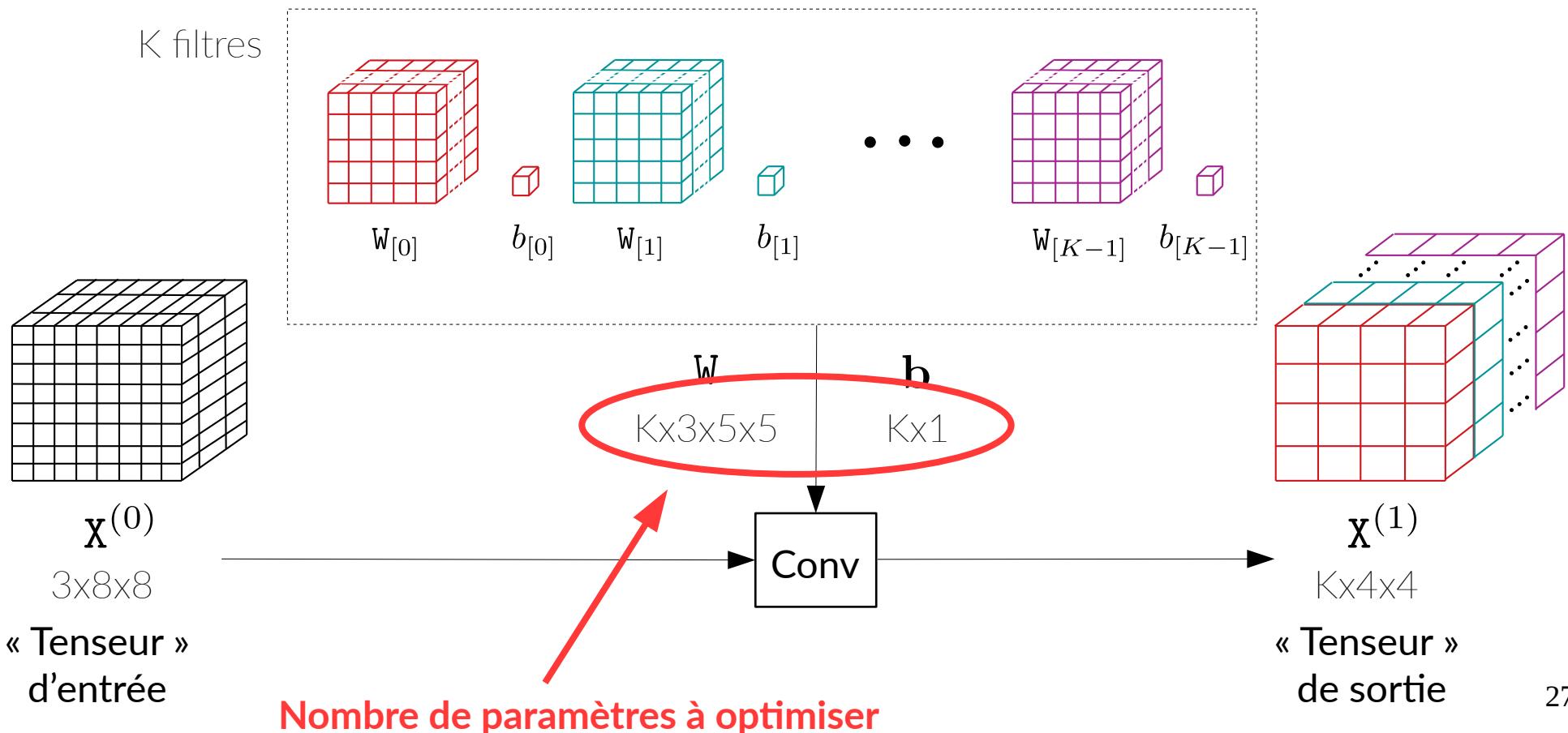
Couche de convolution 2D : K filtres



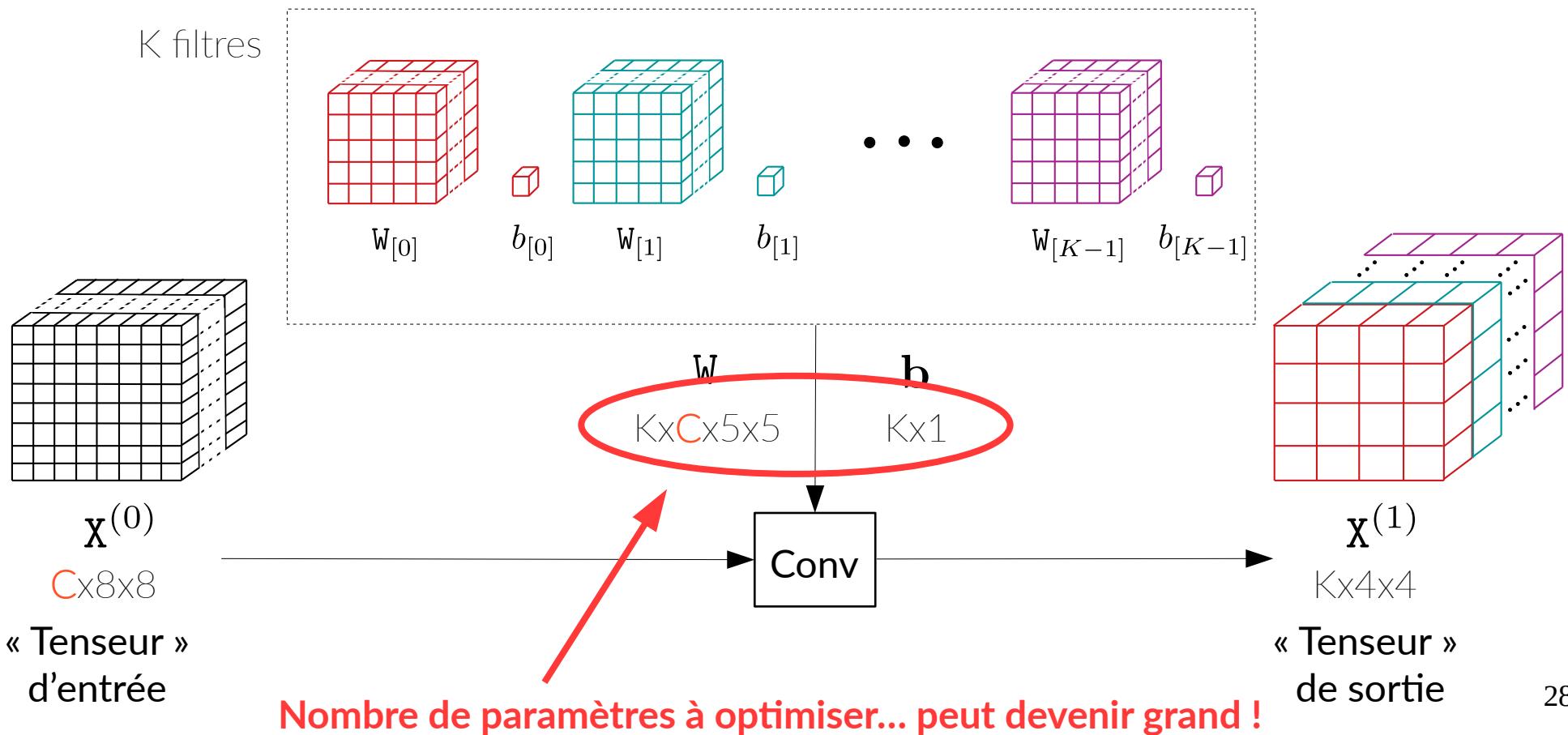
Couche de convolution 2D : K filtres



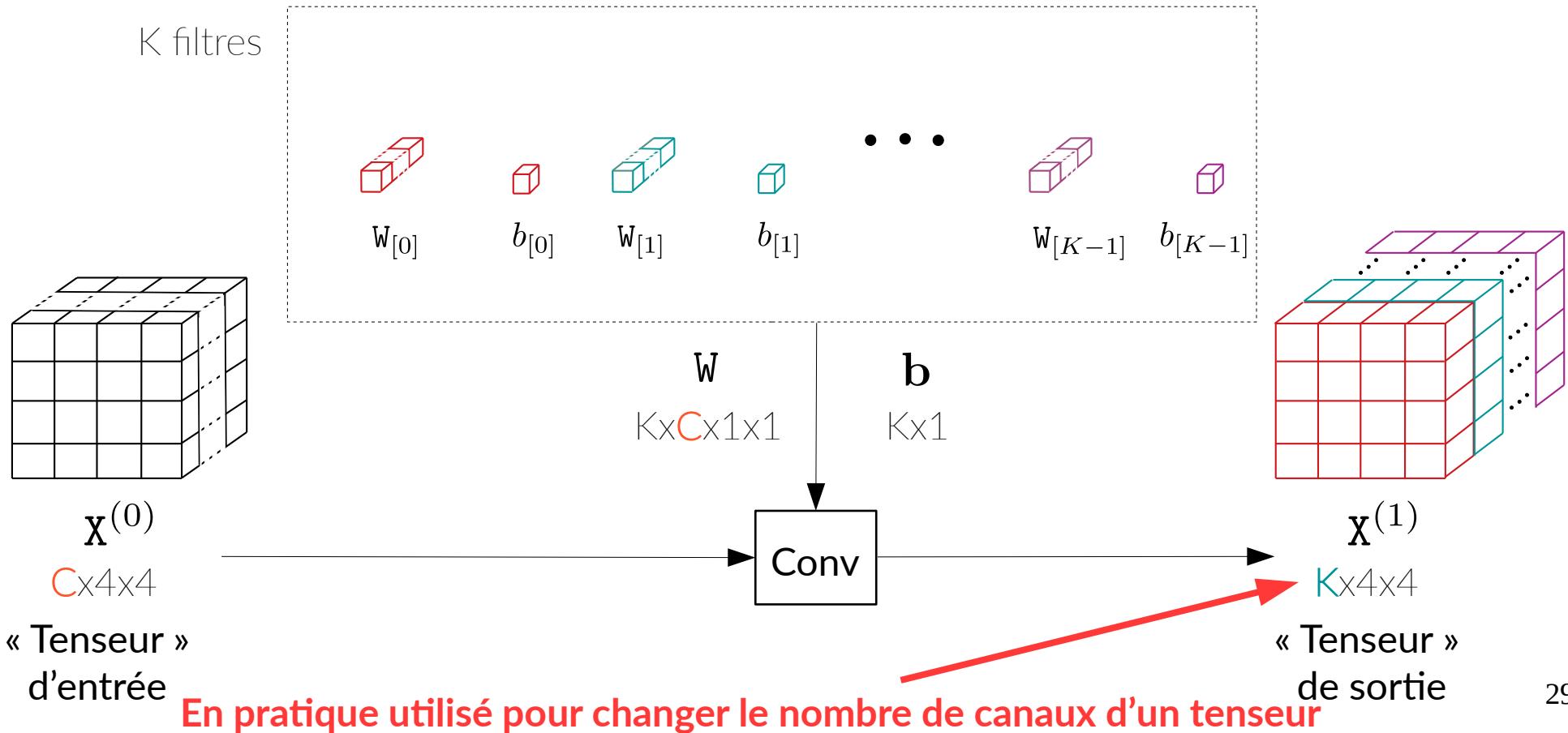
Couche de convolution 2D : K filtres



Couche de convolution 2D : K filtres



Couche de convolution 2D avec filtre 1x1



« Tenseur »
d'entrée

En pratique utilisé pour changer le nombre de canaux d'un tenseur

I)

« Zero padding »

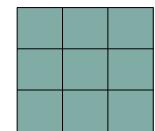
$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & & & & & & & 0 \\ 0 & & & & & & & 0 \\ 0 & & & & & & & 0 \\ 0 & & & & & & & 0 \\ 0 & & & & & & & 0 \\ 0 & & & & & & & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \quad * \quad \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} \quad = \quad \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & & & & & & & & 0 \\ 0 & & & & & & & & 0 \\ 0 & & & & & & & & 0 \\ 0 & & & & & & & & 0 \\ 0 & & & & & & & & 0 \\ 0 & & & & & & & & 0 \\ 0 & & & & & & & & 0 \end{matrix}$$

I)

« Zero padding »

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

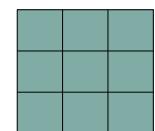
*



=

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

*



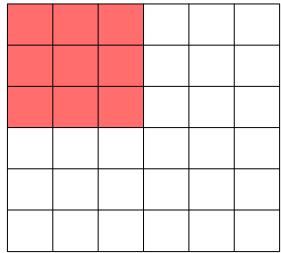
=

Permet de préserver la taille du tenseur d'entrée

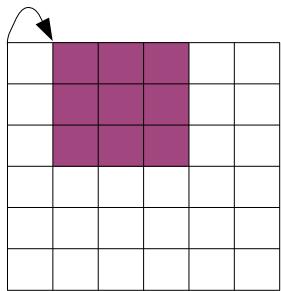
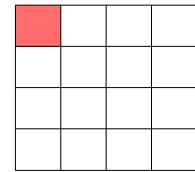
I)

« Stride »

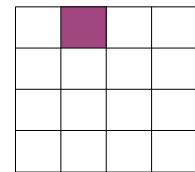
Exemple Stride = 1



* =



* =



« Stride »

Exemple Stride = 1

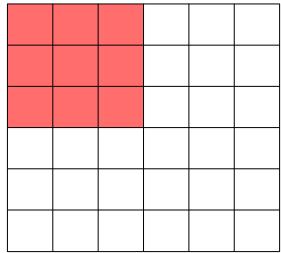
$$\begin{array}{|c|c|c|c|c|c|} \hline & & & & & \\ \hline \end{array} \times \begin{array}{|c|c|c|c|c|c|} \hline & & & & & \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|} \hline & & & & & \\ \hline \end{array}$$

Exemple Stride = 2

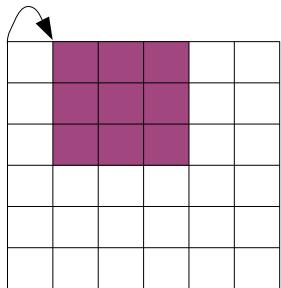
The diagram illustrates the multiplication of a 3x3 matrix by a 3x1 column vector. On the left, a 3x3 grid of squares is shown, with the first three columns shaded orange. A curved arrow points from the top-left white square to the first column of orange squares. To the right of this grid is a multiplication sign (*). To the right of the multiplication sign is a 3x1 column vector consisting of three teal squares stacked vertically. To the right of the column vector is an equals sign (=). To the right of the equals sign is a 3x1 column vector consisting of two white squares and one orange square.

« Stride »

Exemple Stride = 1

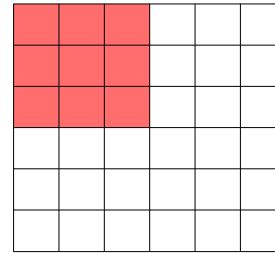


$$\begin{matrix} * & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} & = & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} \end{matrix}$$

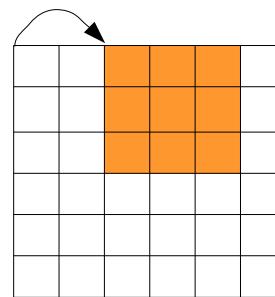


$$\begin{matrix} * & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} & = & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} \end{matrix}$$

Exemple Stride = 2



$$\begin{matrix} * & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} & = & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} \end{matrix}$$



$$\begin{matrix} * & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} & = & \begin{matrix} \text{---} \\ \text{---} \\ \text{---} \end{matrix} \end{matrix}$$

Réduit la résolution de la sortie par « stride »

Principaux hyperparamètres d'une couche de convolution

- Taille des filtres
 - En pratique toujours impair
 - Souvent 1x1 ou 3x3, parfois 5x5 ou 7x7

Principaux hyperparamètres d'une couche de convolution

- Taille des filtres
 - En pratique toujours impair
 - Souvent 1x1 ou 3x3, parfois 5x5 ou 7x7
- Nombre de filtres
 - = Nombre de canaux souhaité en sortie

Principaux hyperparamètres d'une couche de convolution

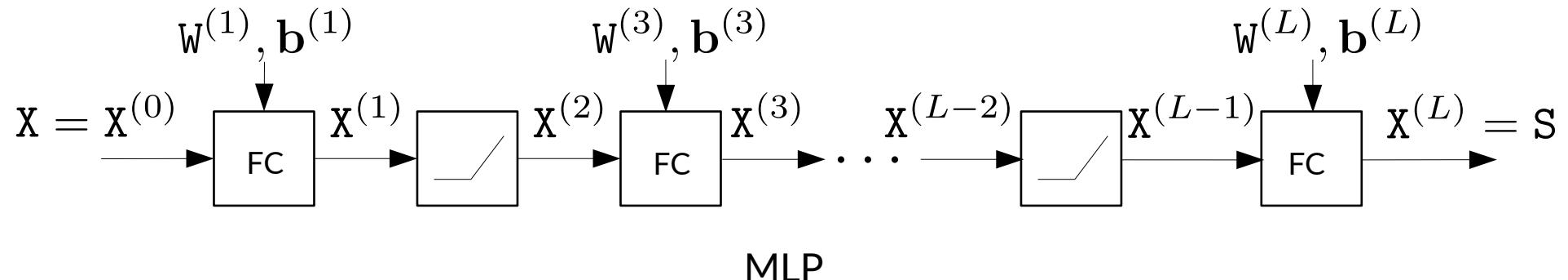
- Taille des filtres
 - En pratique toujours impair
 - Souvent 1x1 ou 3x3, parfois 5x5 ou 7x7
- Nombre de filtres
 - = Nombre de canaux souhaité en sortie
- Quantité de zero-padding
 - Compense la taille du filtre si volonté de préserver la taille de l'entrée
 - $1 \times 1 \rightarrow \text{padding} = 0$, $3 \times 3 \rightarrow \text{padding} = 1$, $5 \times 5 \rightarrow \text{padding} = 2$

Principaux hyperparamètres d'une couche de convolution

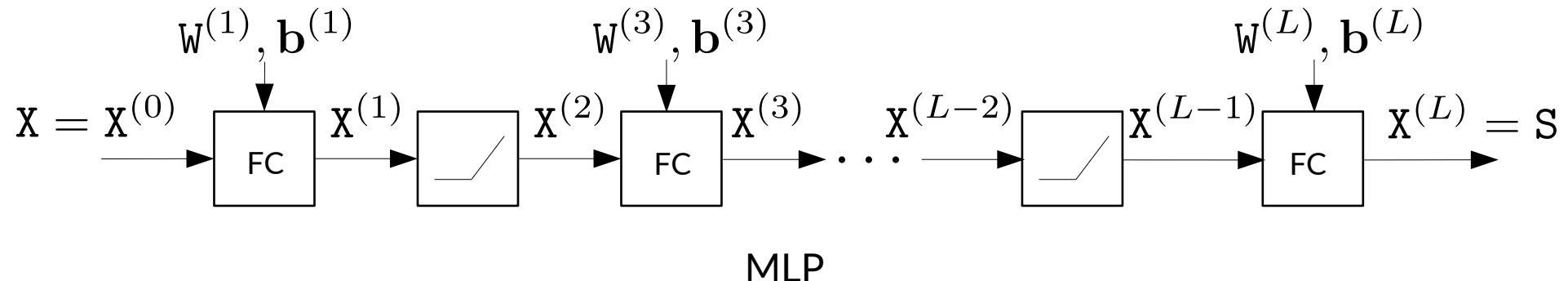
- Taille des filtres
 - En pratique toujours impair
 - Souvent 1x1 ou 3x3, parfois 5x5 ou 7x7
- Nombre de filtres
 - = Nombre de canaux souhaité en sortie
- Quantité de zero-padding
 - Compense la taille du filtre si volonté de préserver la taille de l'entrée
 - $1 \times 1 \rightarrow \text{padding} = 0$, $3 \times 3 \rightarrow \text{padding} = 1$, $5 \times 5 \rightarrow \text{padding} = 2$
- Stride
 - = 1 si volonté de préserver la résolution de l'entrée
 - = 2 si volonté de réduire la résolution de l'entrée

II) Réseau de neurones à convolution

Réseau de neurones à convolution (CNN)



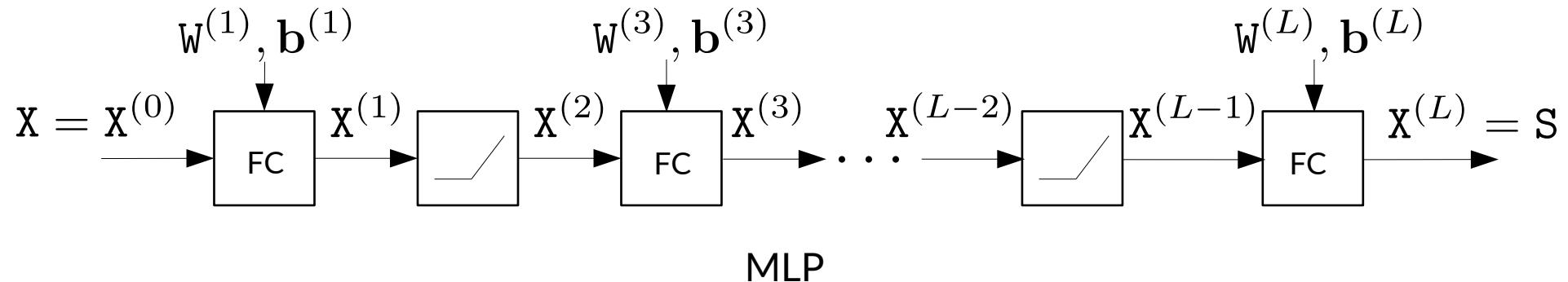
Réseau de neurones à convolution (CNN)



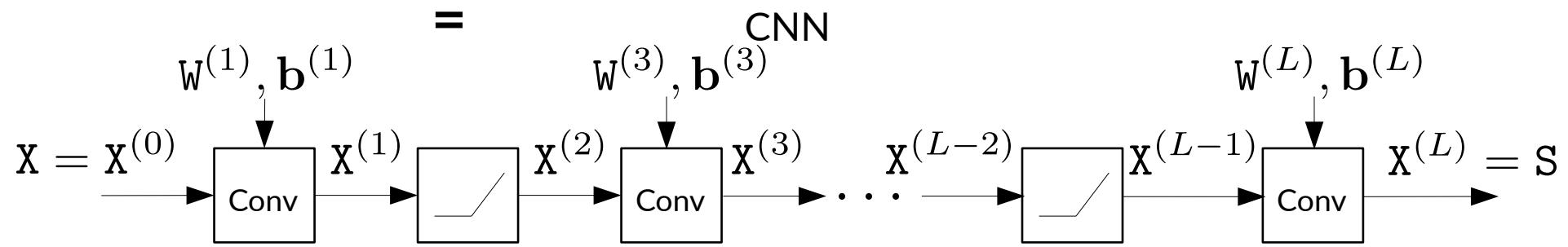
MLP

- FC (transformations affines générales)
 - + Conv (transformations affines spécifiques)
- = CNN

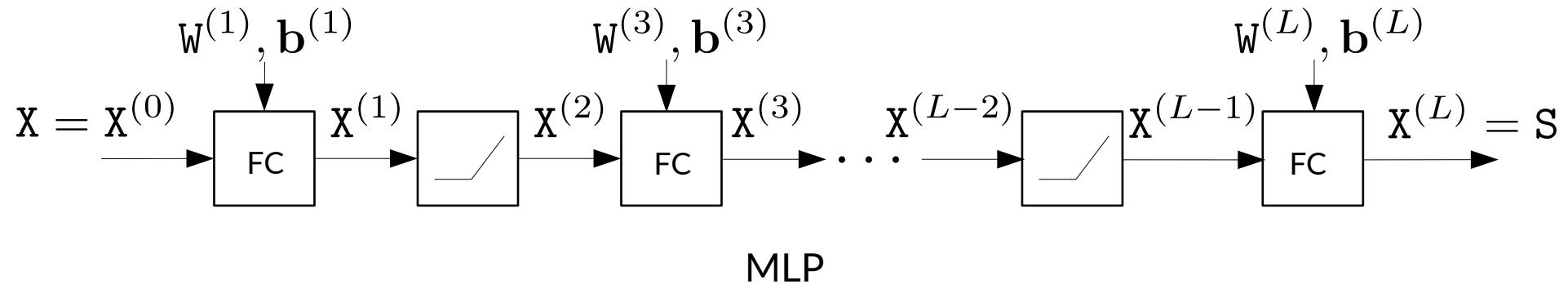
Réseau de neurones à convolution (CNN)



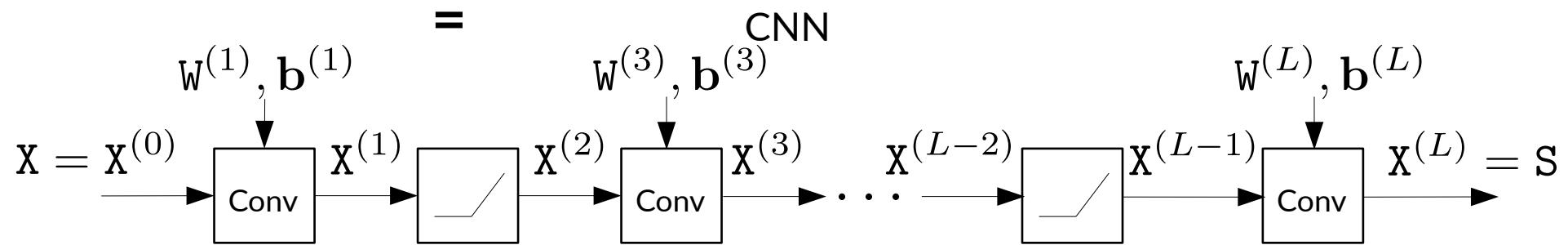
- FC (transformations affines générales)
- + Conv (transformations affines spécifiques)



Réseau de neurones à convolution (CNN)

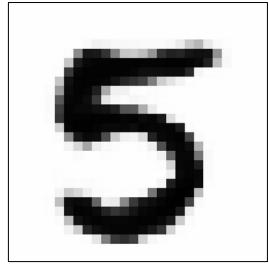


- FC (transformations affines générales)
- + Conv (transformations affines spécifiques)

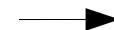


→ Initialisation des paramètres d'une couche de convolution identique à ceux d'une FC !

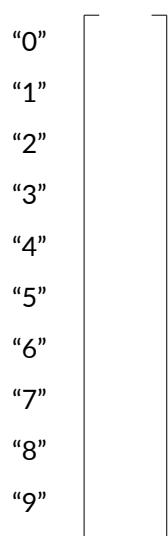
Exemple d'architecture de CNN pour MNIST



$$x = x^{(0)}$$



1x28x28

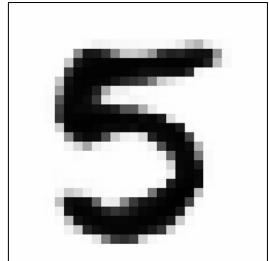


S

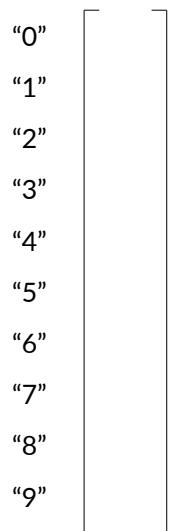
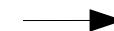
10x1x1



Exemple d'architecture de CNN pour MNIST



$$x = x^{(0)}$$



1x28x28

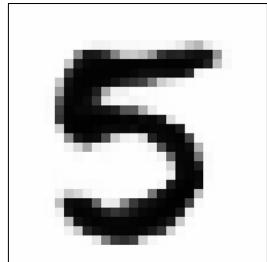
Objectif : réduire progressivement la résolution de 28x28 à 1x1 et agrandir le nombre de canaux de 1 à 10.

S

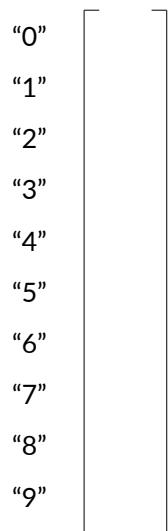
10x1x1



Exemple d'architecture de CNN pour MNIST



$$x = x^{(0)}$$



1x28x28

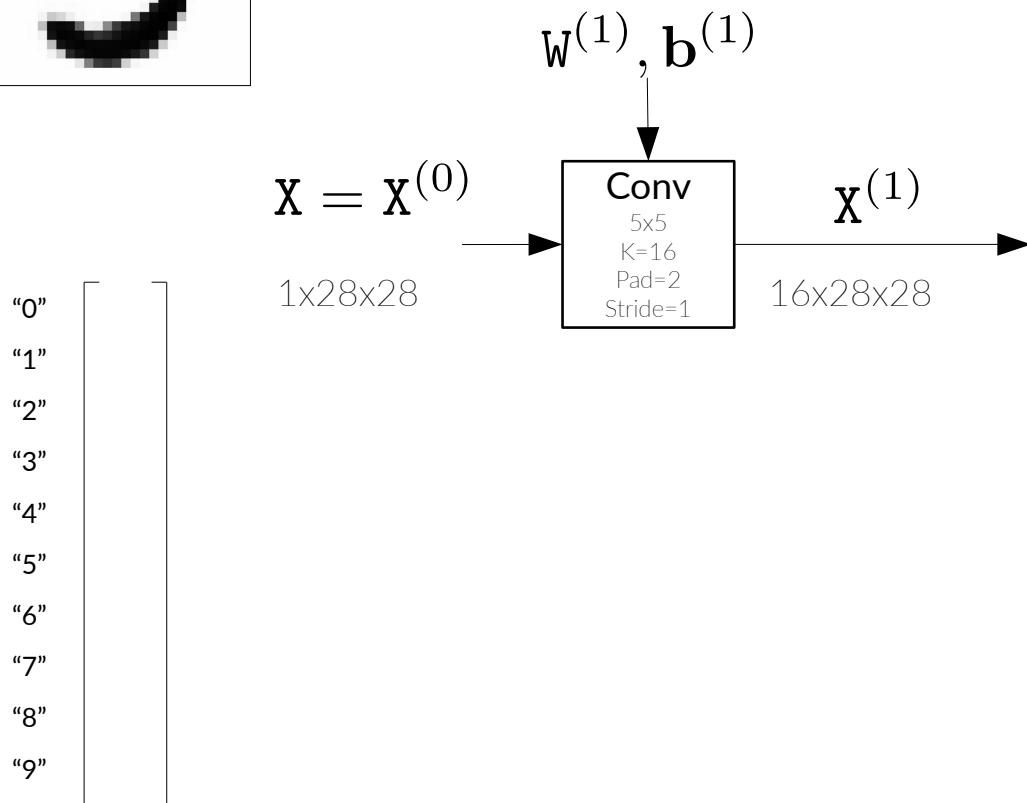
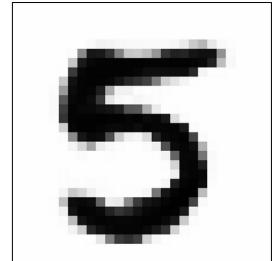
Objectif : réduire progressivement la résolution de 28x28 à 1x1 et agrandir le nombre de canaux de 1 à 10.

S

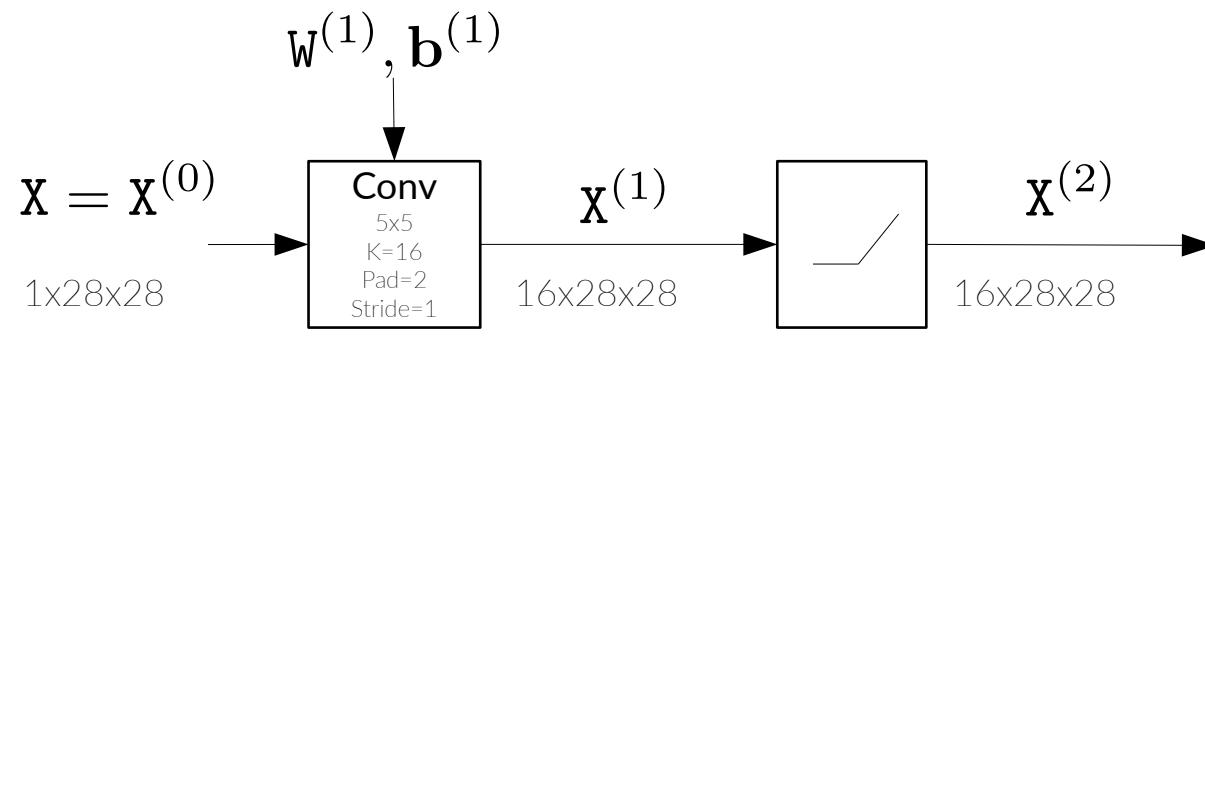
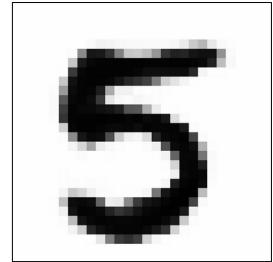
10x1x1

Remarque : en général on augmente beaucoup plus le nombre de canaux au sein du réseau pour ne pas « trop » compresser l'information.

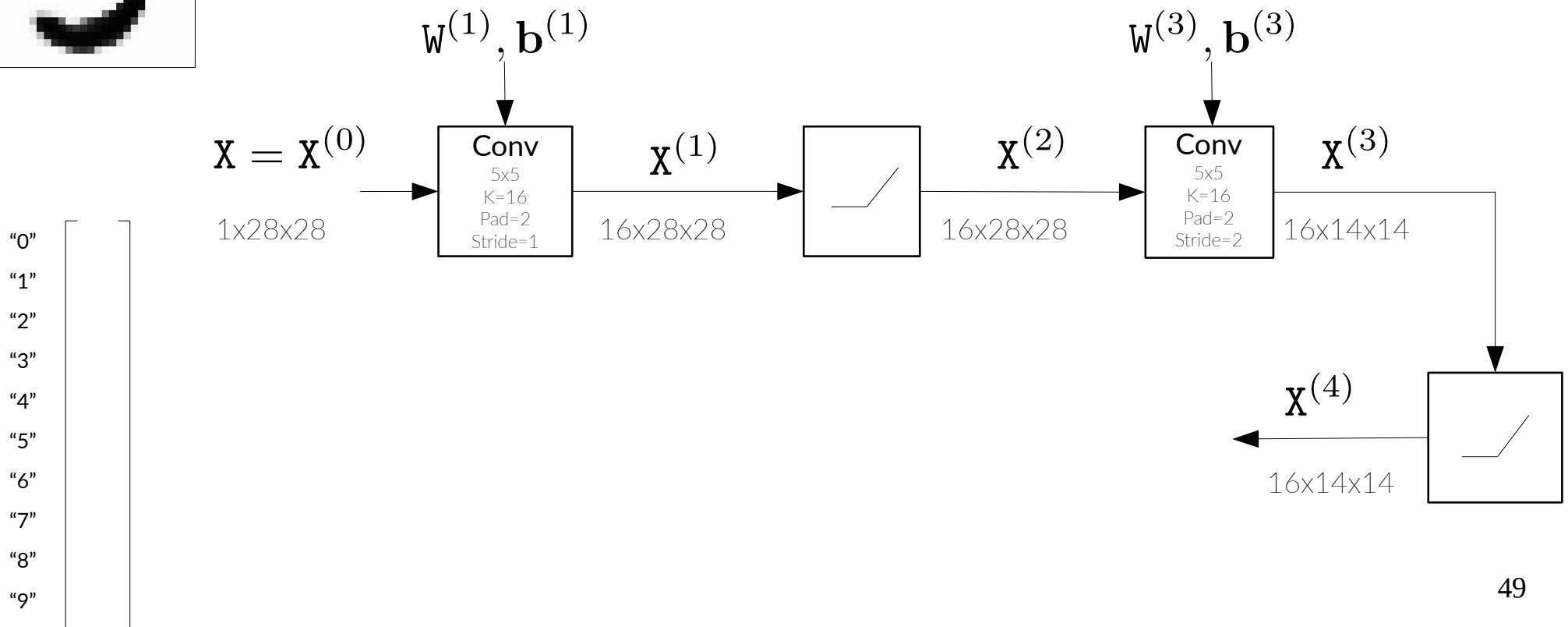
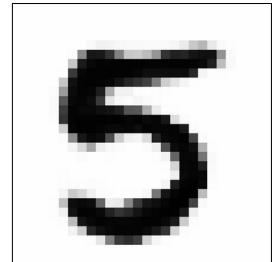
Exemple d'architecture de CNN pour MNIST



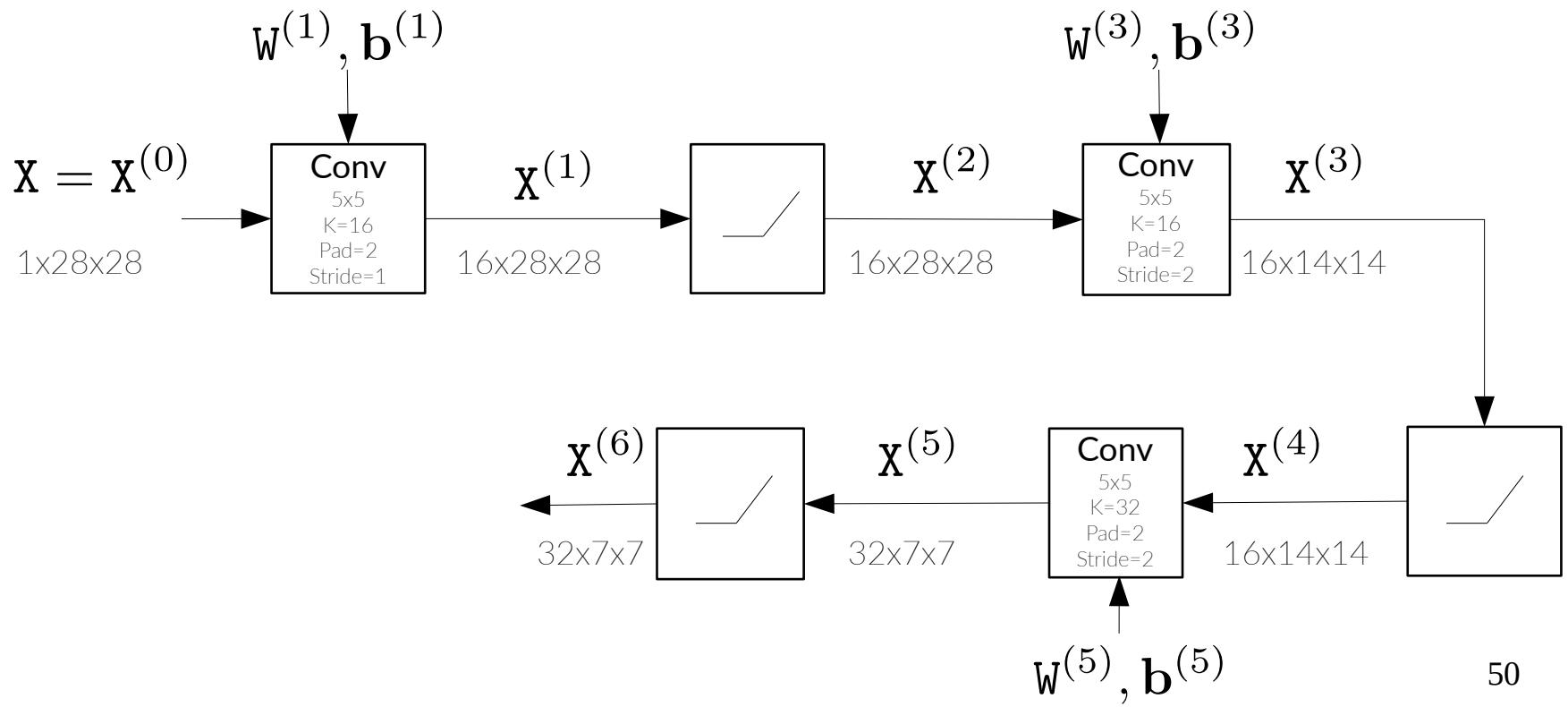
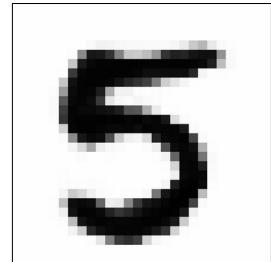
Exemple d'architecture de CNN pour MNIST



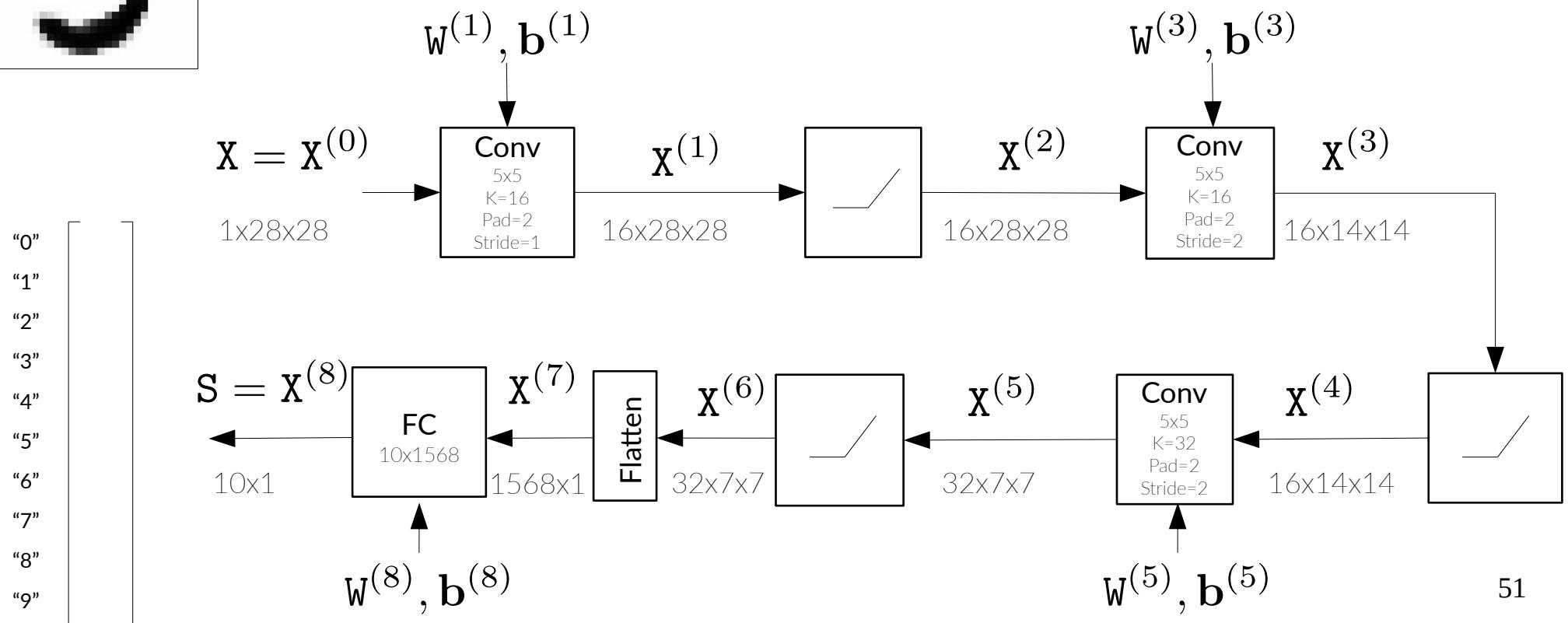
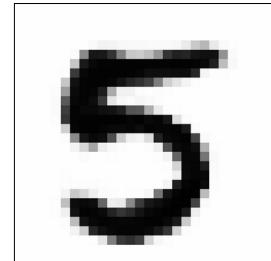
Exemple d'architecture de CNN pour MNIST



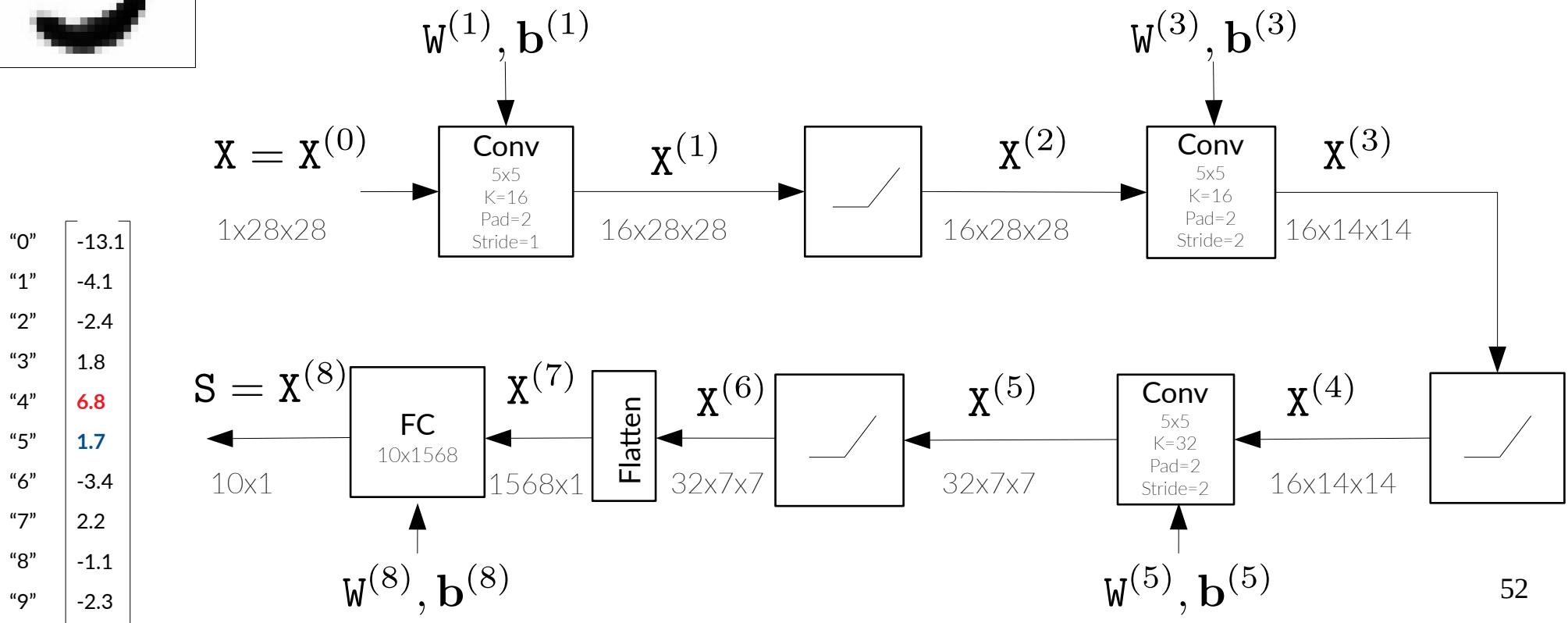
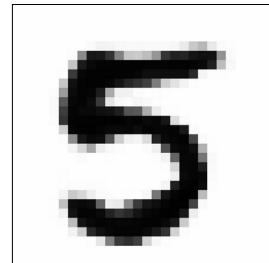
Exemple d'architecture de CNN pour MNIST



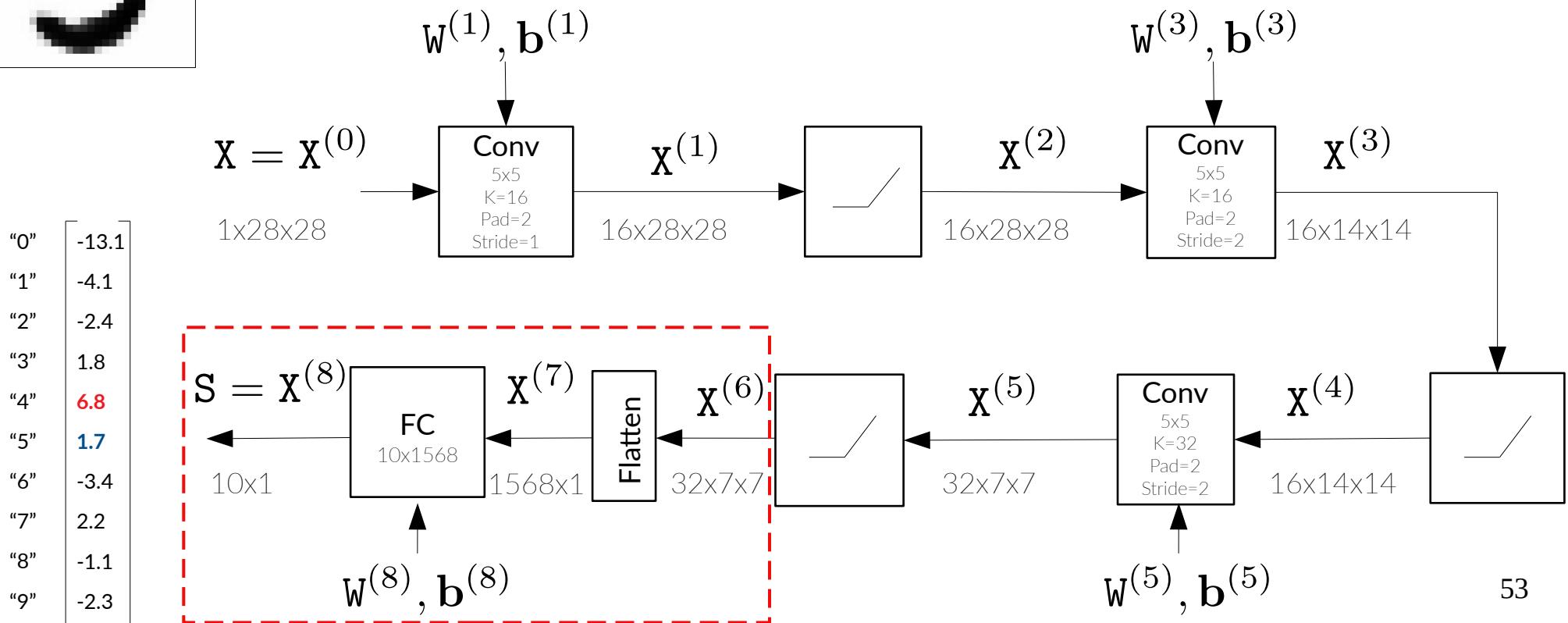
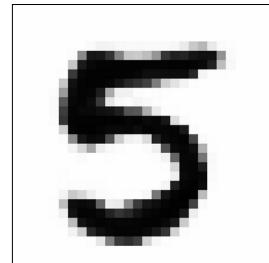
Exemple d'architecture de CNN pour MNIST



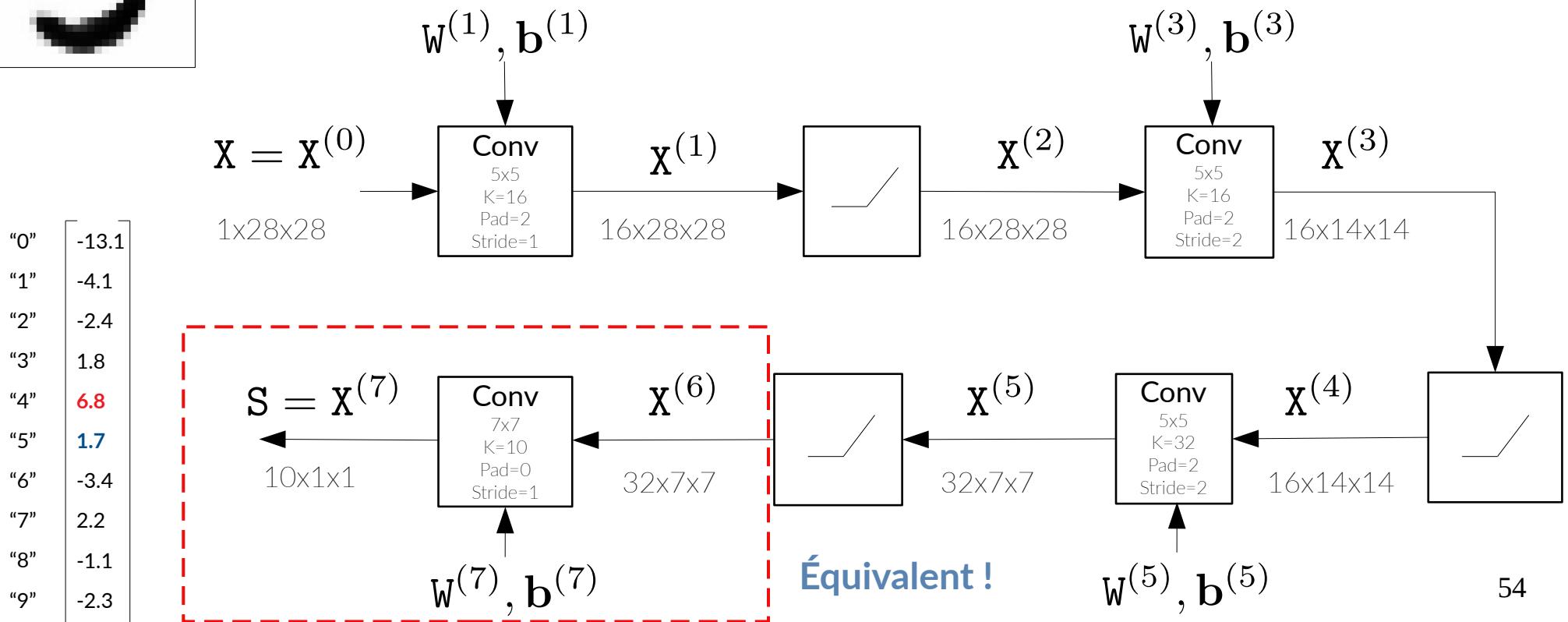
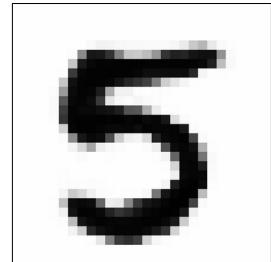
Exemple d'architecture de CNN pour MNIST



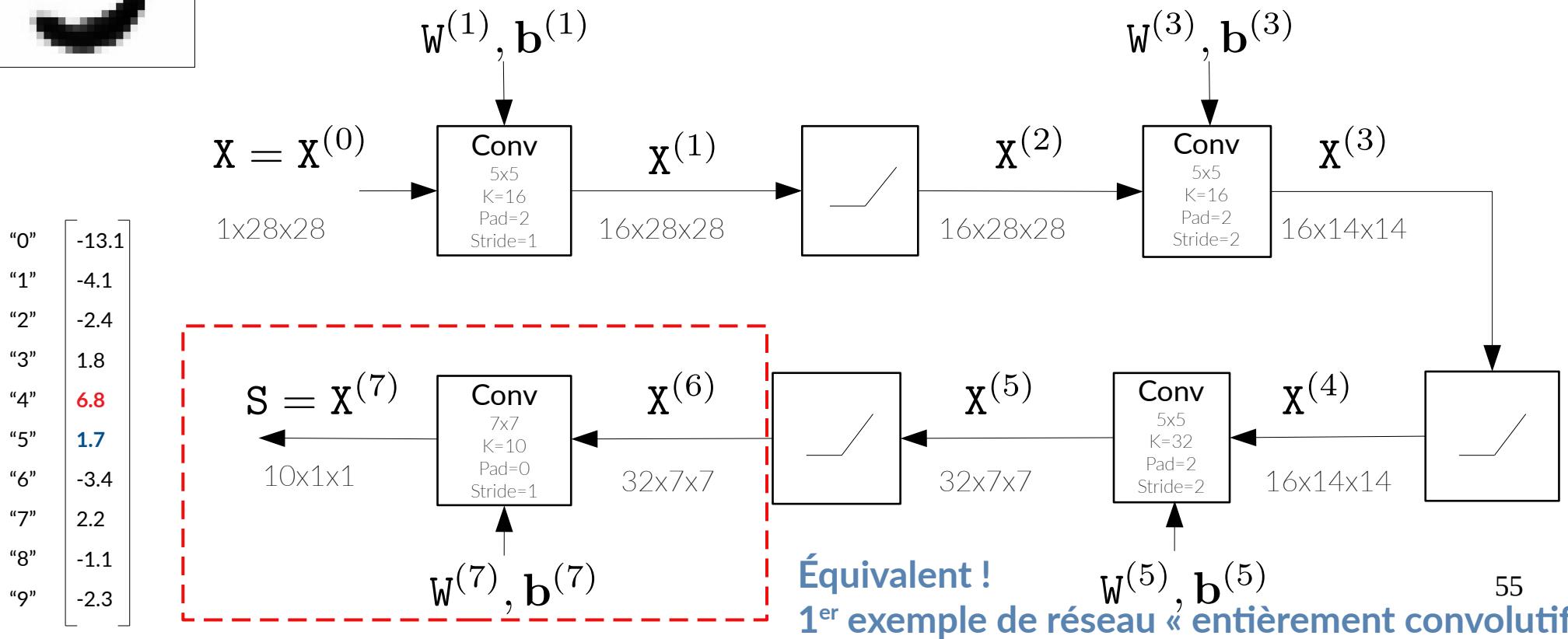
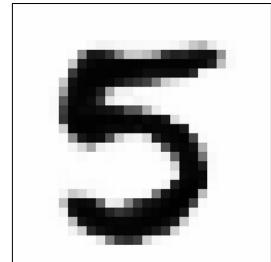
Exemple d'architecture de CNN pour MNIST



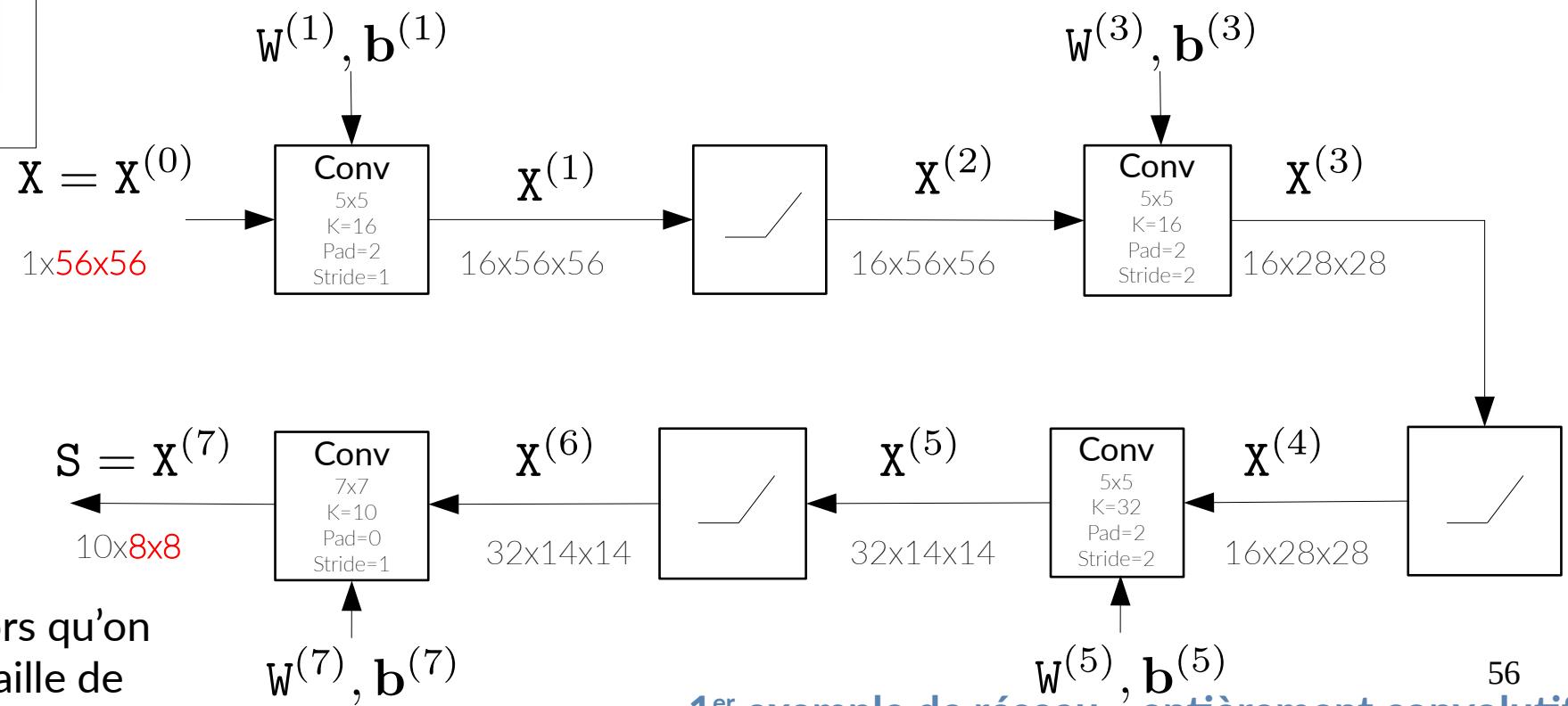
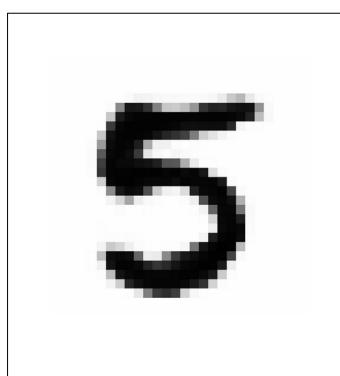
Exemple d'architecture de CNN pour MNIST



Exemple d'architecture de CNN pour MNIST



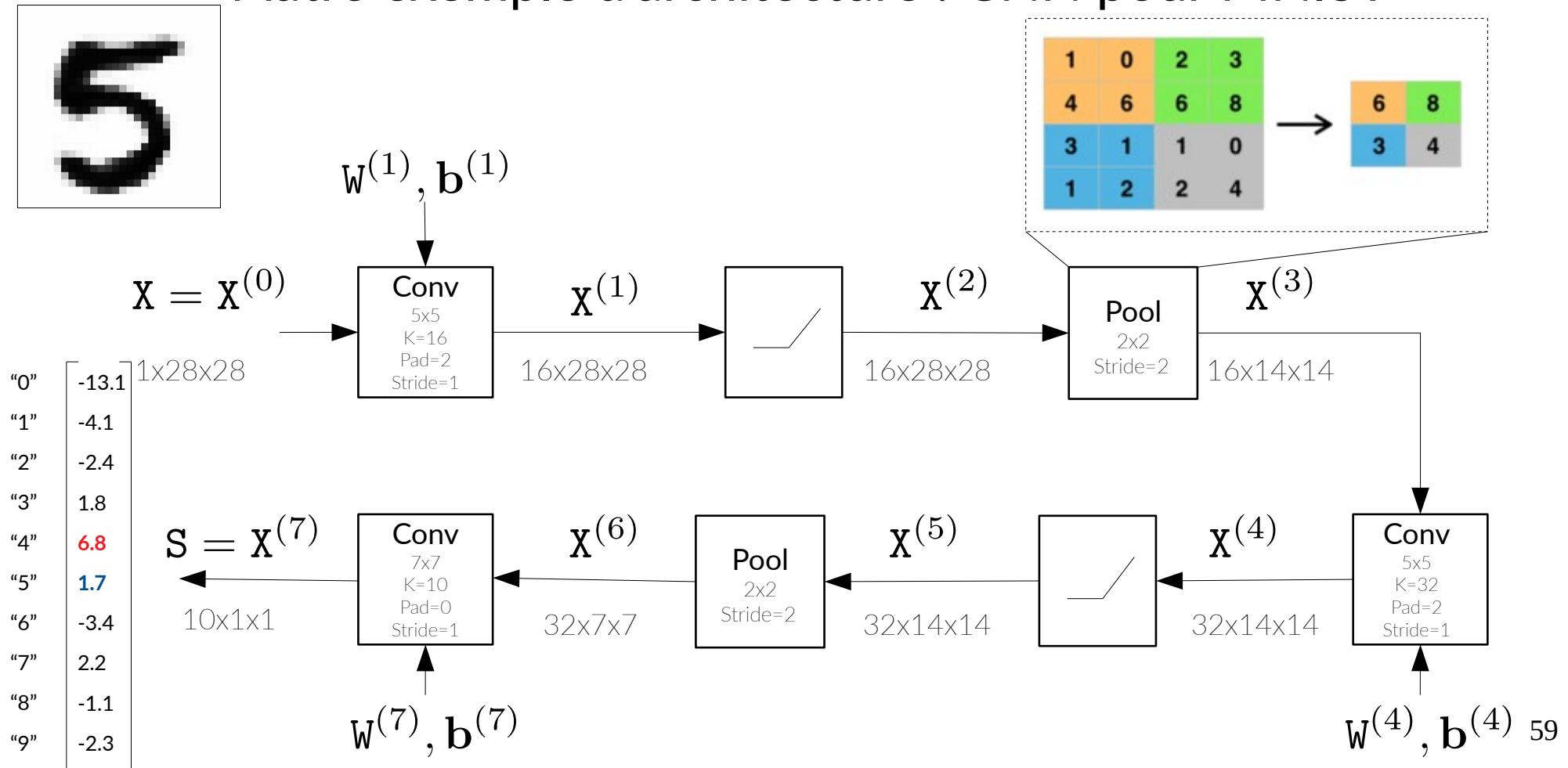
Exemple d'architecture de CNN pour MNIST



S'exécute alors qu'on a changé la taille de l'entrée

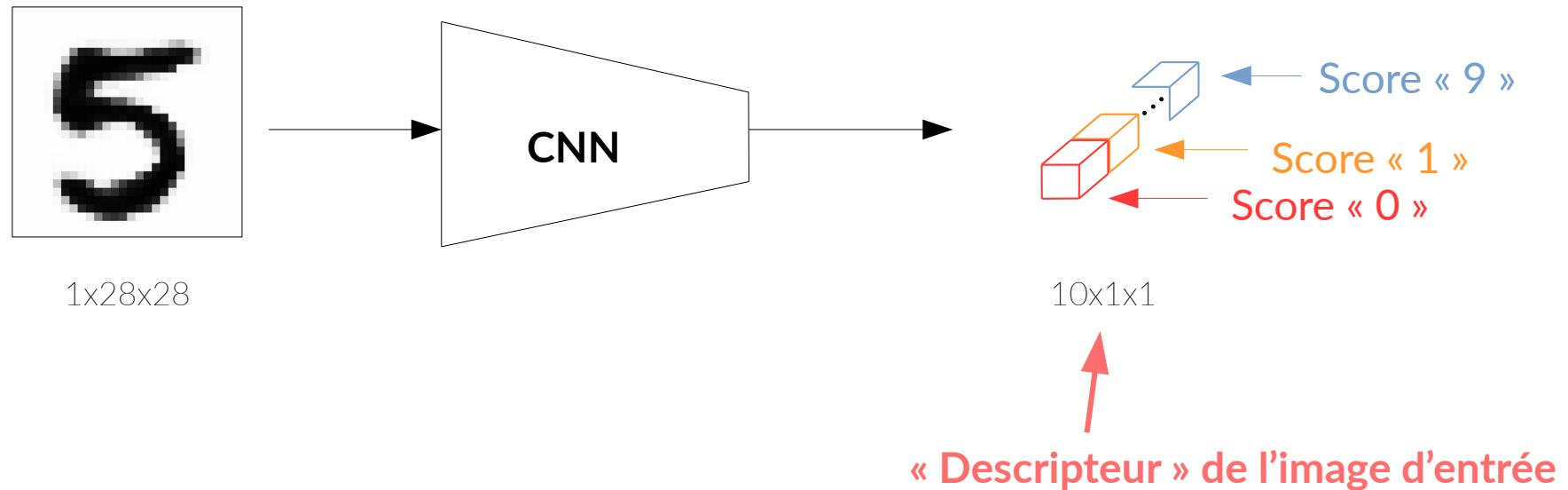
II)

Autre exemple d'architecture : CNN pour MNIST



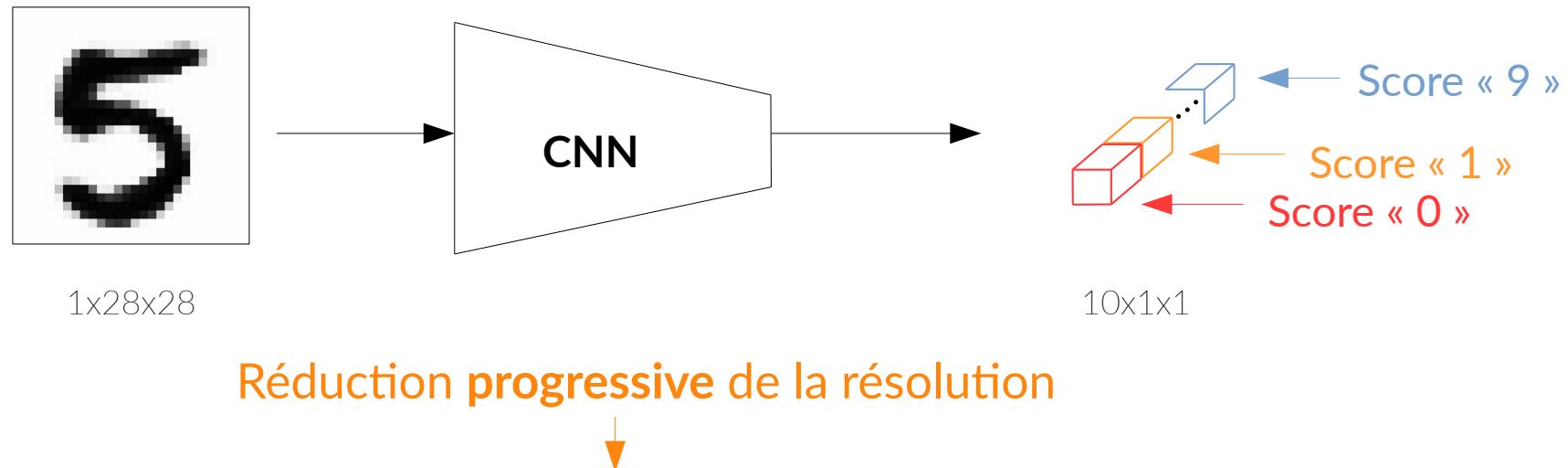
Architectures de CNN : Deux cas extrêmes

Cas 1 : Extraire une information **globale** présente dans l'image d'entrée



Architectures de CNN : Deux cas extrêmes

Cas 1 : Extraire une information **globale** présente dans l'image d'entrée



Utilisation de couches de conv ou pooling avec stride = 2

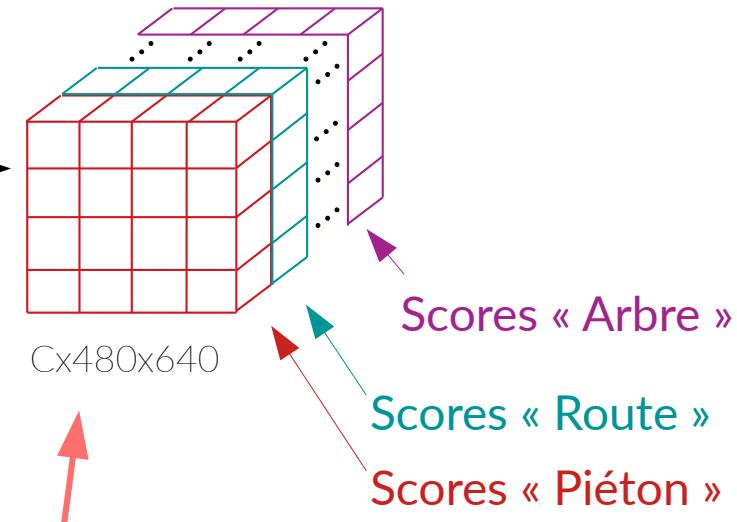
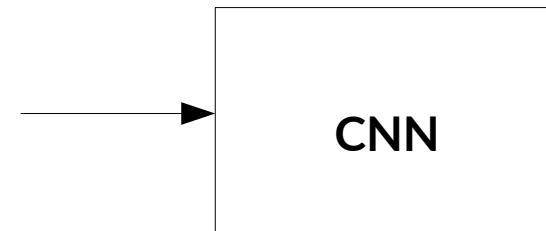
Architectures de CNN : Deux cas extrêmes

Cas 1 : Extraire une information **globale** présente dans l'image d'entrée

Cas 2 : Extraire une information pour chaque **pixel** de l'image d'entrée

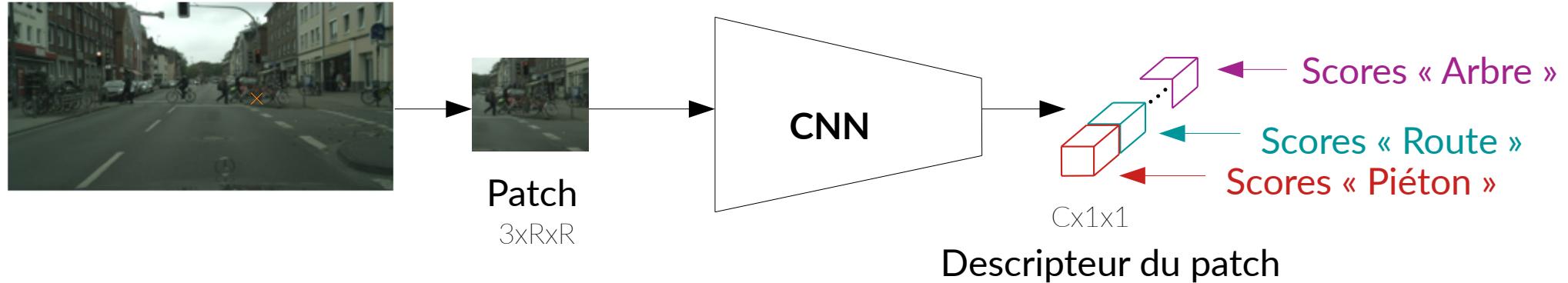


3x480x640

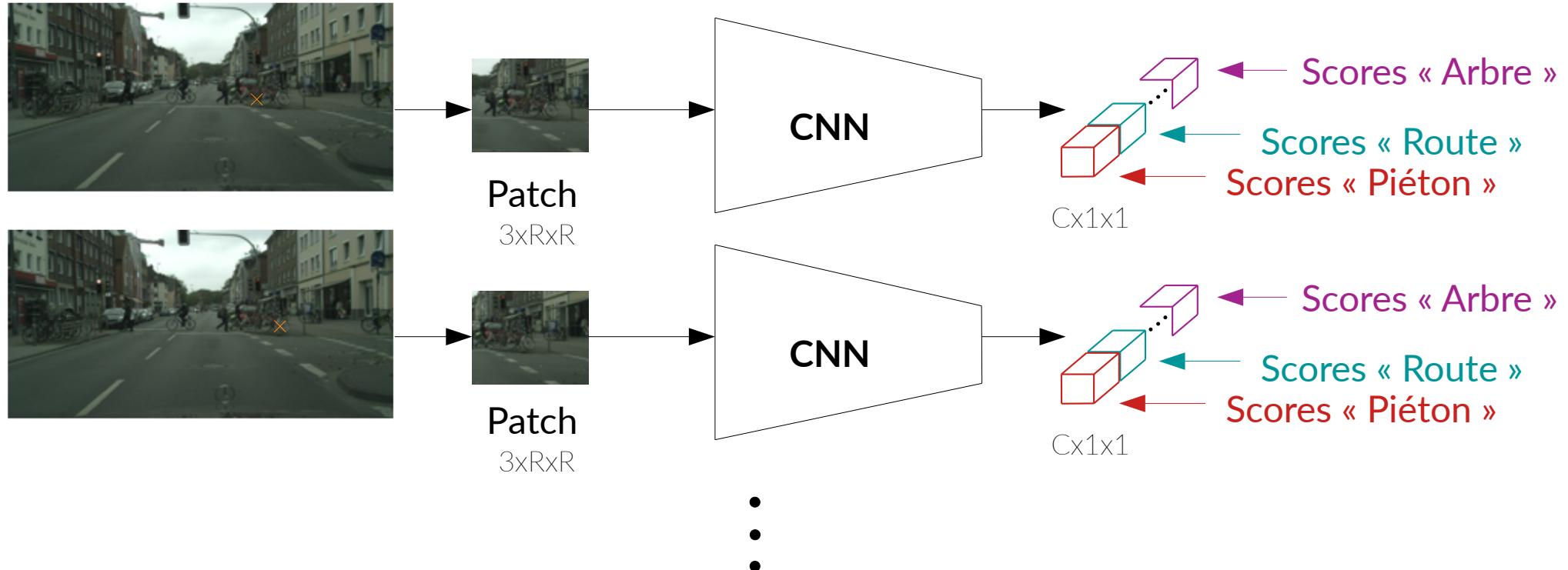


Un « descripteur » par pixel de l'image d'entrée

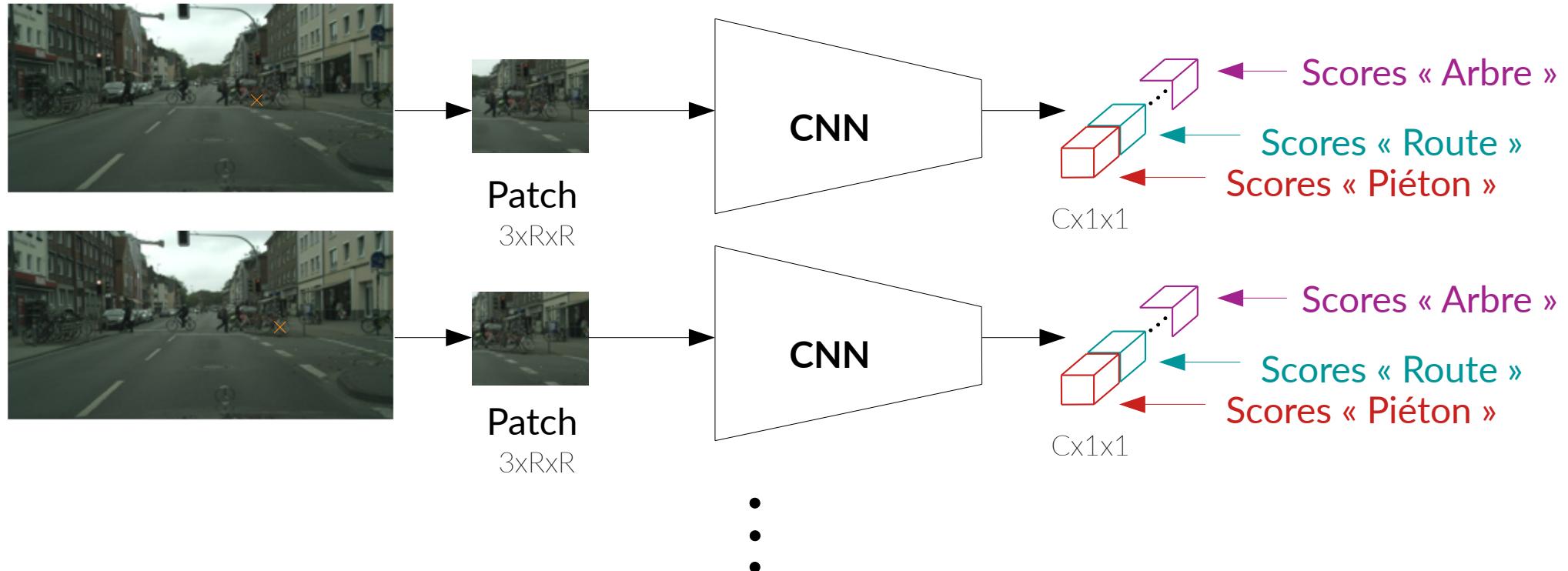
Comment obtenir un descripteur pour chaque pixel ?



Comment obtenir un descripteur pour chaque pixel ?



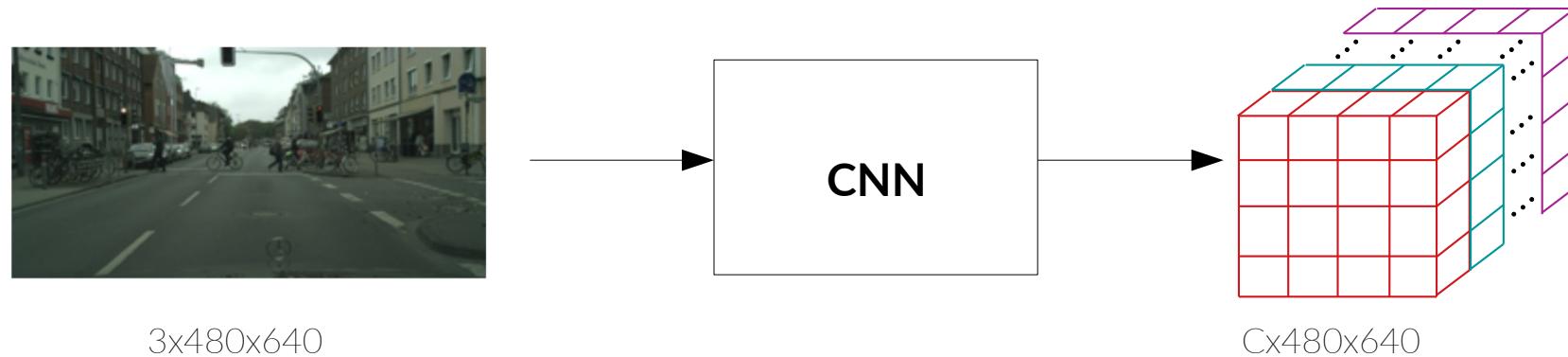
Comment obtenir un descripteur pour chaque pixel ?



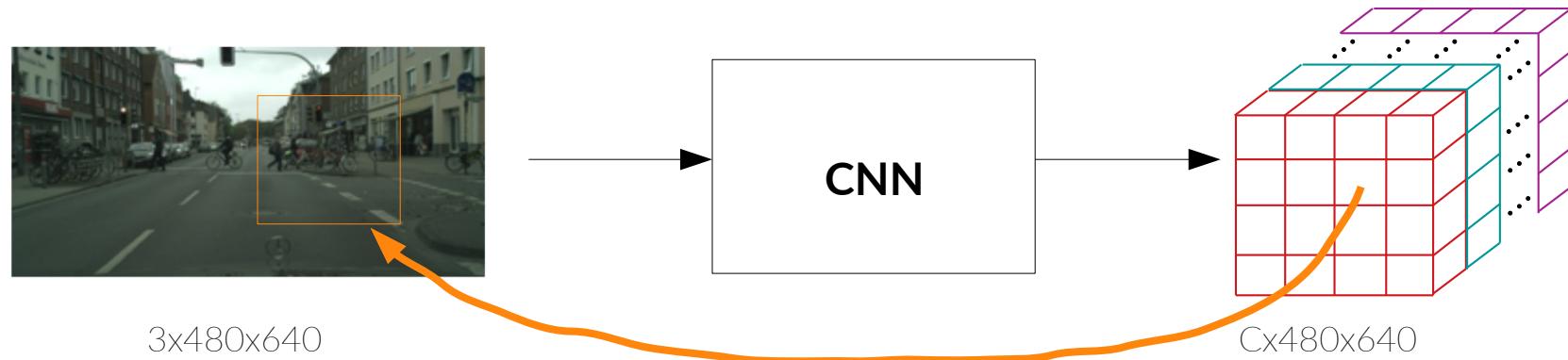
Les patchs voisins ont beaucoup de pixels en commun → calculs redondants

→ Utilisation d'une architecture entièrement convolutive

Architecture entièrement convolutive

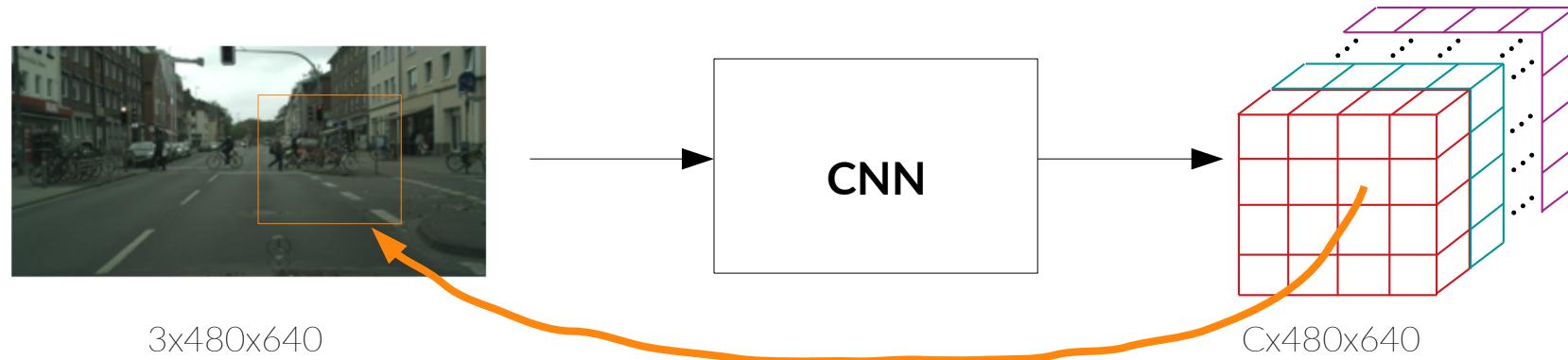


Architecture entièrement convolutive



Descripteur $C \times 1 \times 1$ d'un patch de taille $R \times R$,
où $R \times R$ s'appelle le Champ Récepteur du CNN (« Receptive Field »)

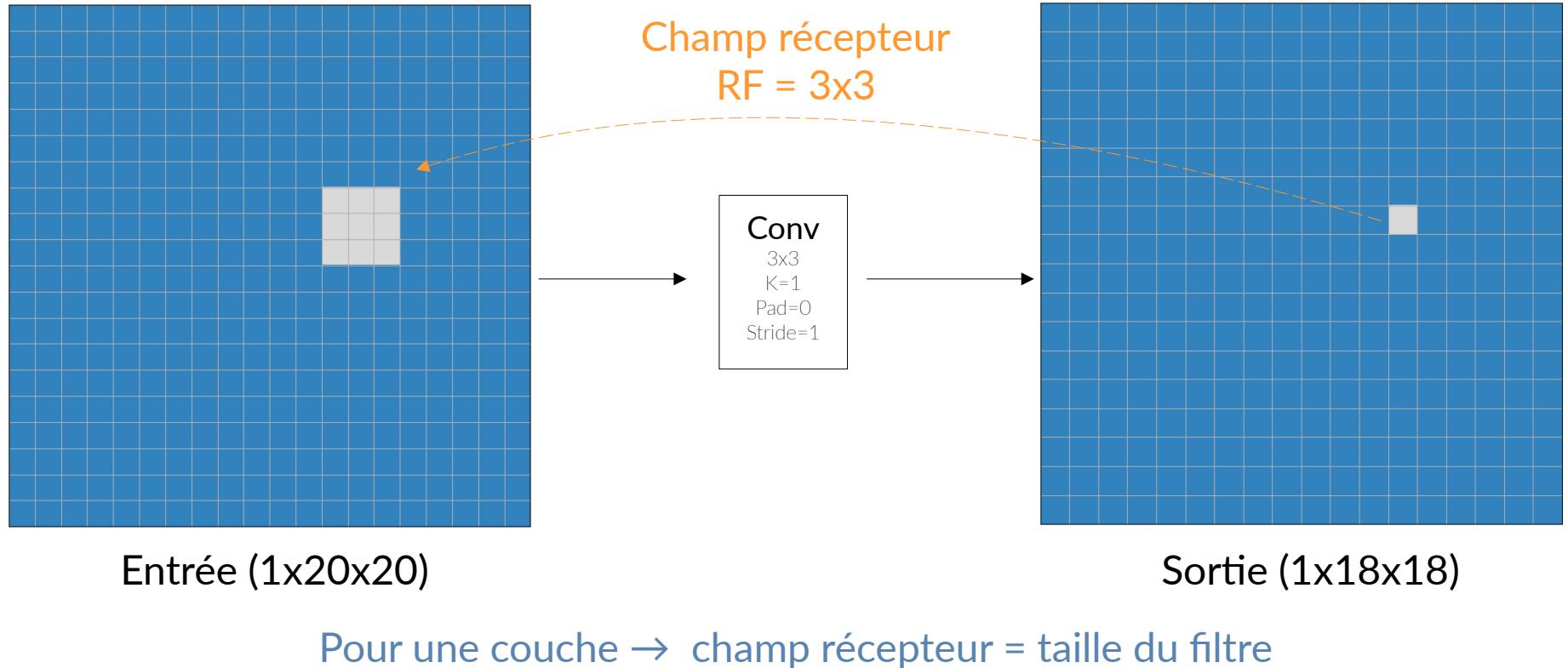
Architecture entièrement convolutive



Descripteur $C \times 1 \times 1$ d'un patch de taille $R \times R$,
où $R \times R$ s'appelle le Champ Récepteur du CNN (« Receptive Field »)

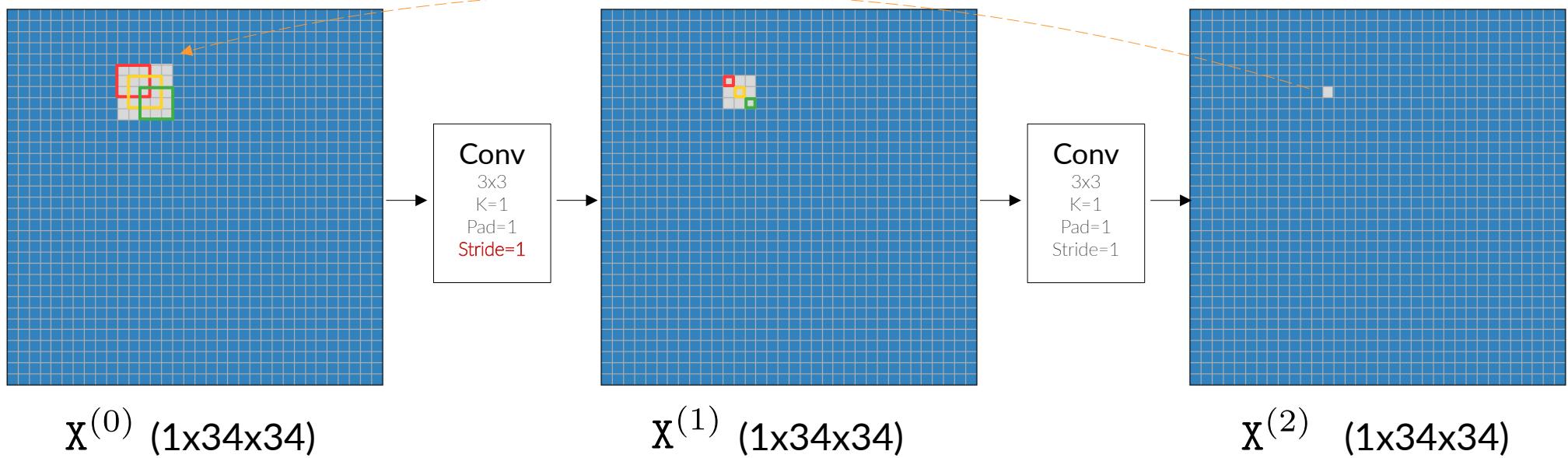
Comment avoir un grand champ récepteur tout en restant
raisonnable en mémoire et temps de calculs ?

Champ récepteur d'une couche de convolution



Champ récepteur de deux couches de convolution Stride = 1

Champ récepteur
RF = 5x5

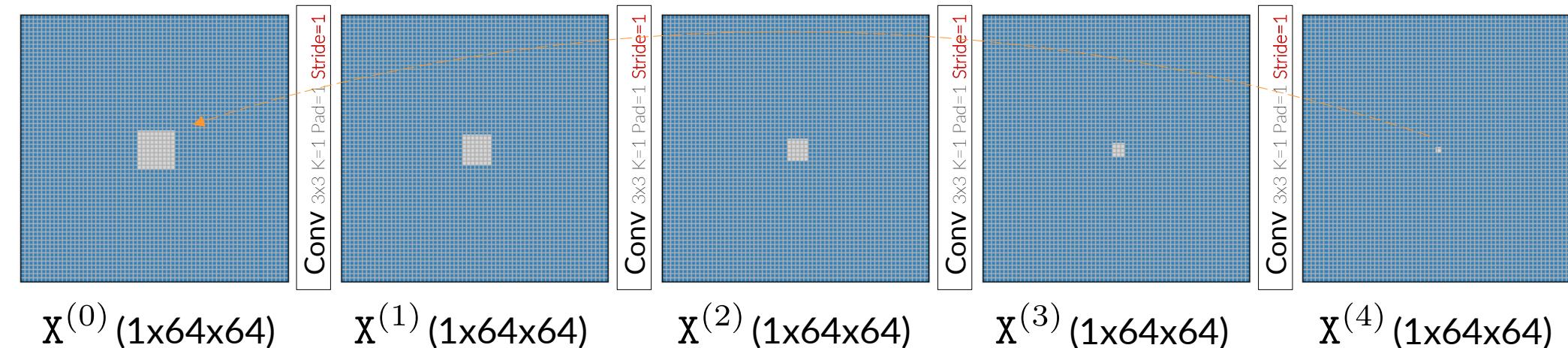


Remarque : les ReLU n'affectent pas le RF donc on ne les représente pas ici.

II)

Champ récepteur de quatre couches de convolution Stride = 1

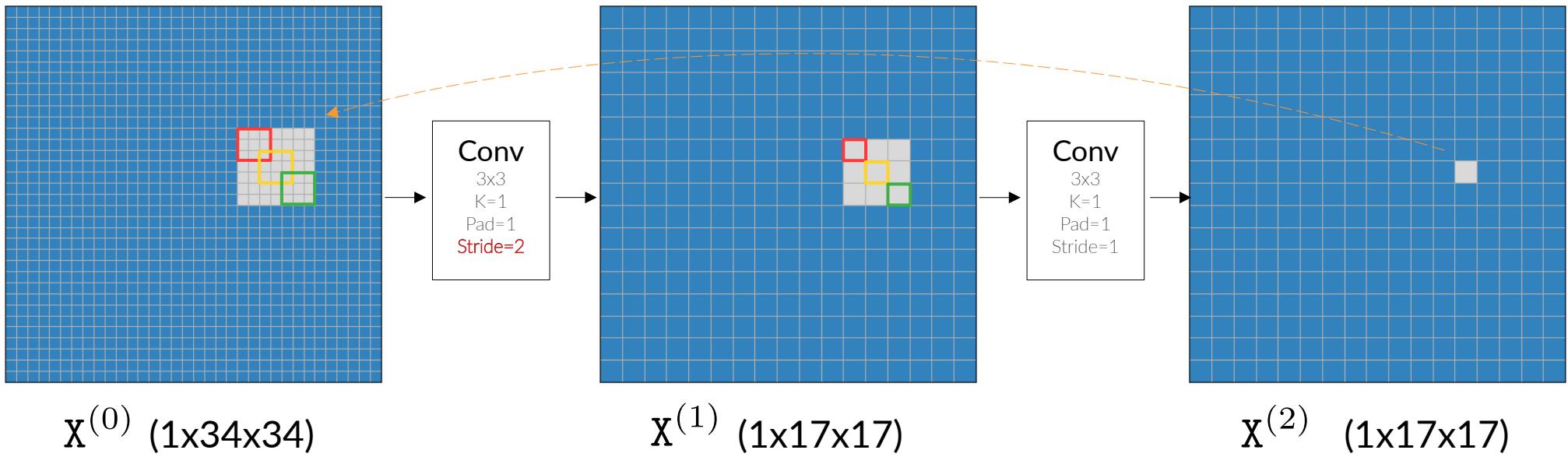
Champ récepteur
RF = 9x9



→ Croissance du RF très lente ...

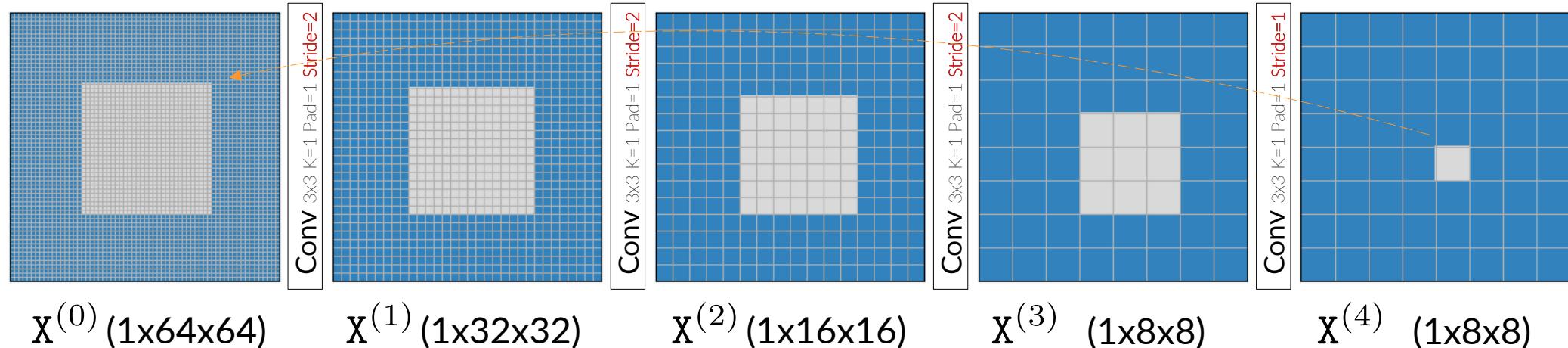
Champ récepteur de deux couches de convolution Stride = 2

Champ récepteur
RF = 7x7



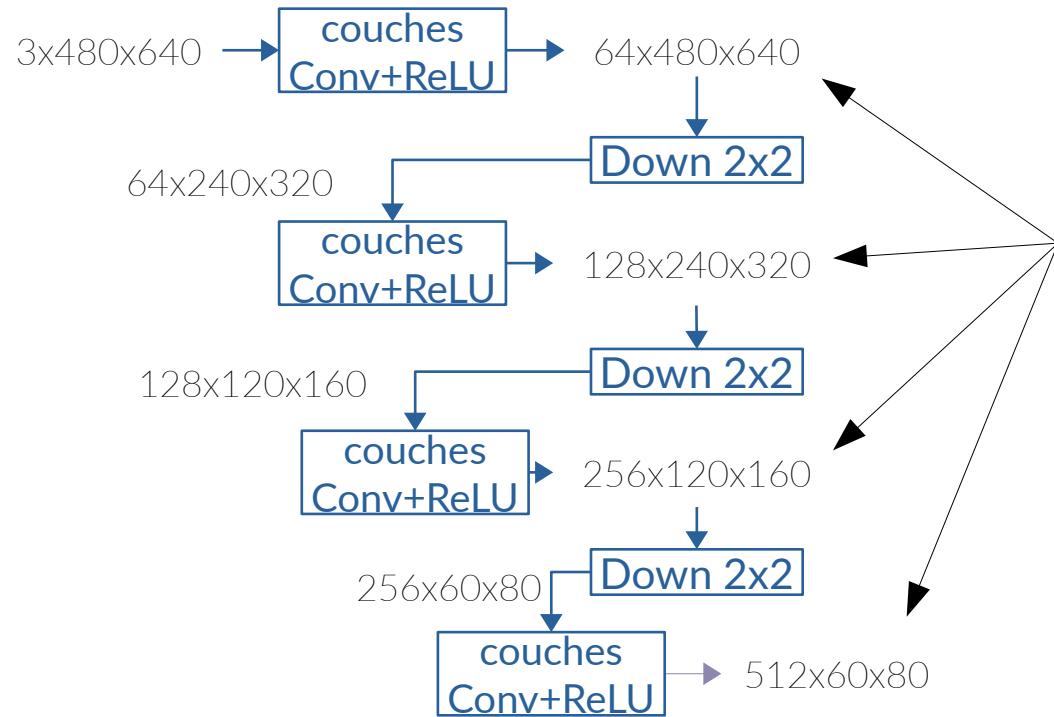
Champ récepteur de quatre couches de convolution Stride = 2

Champ récepteur
RF = 31x31



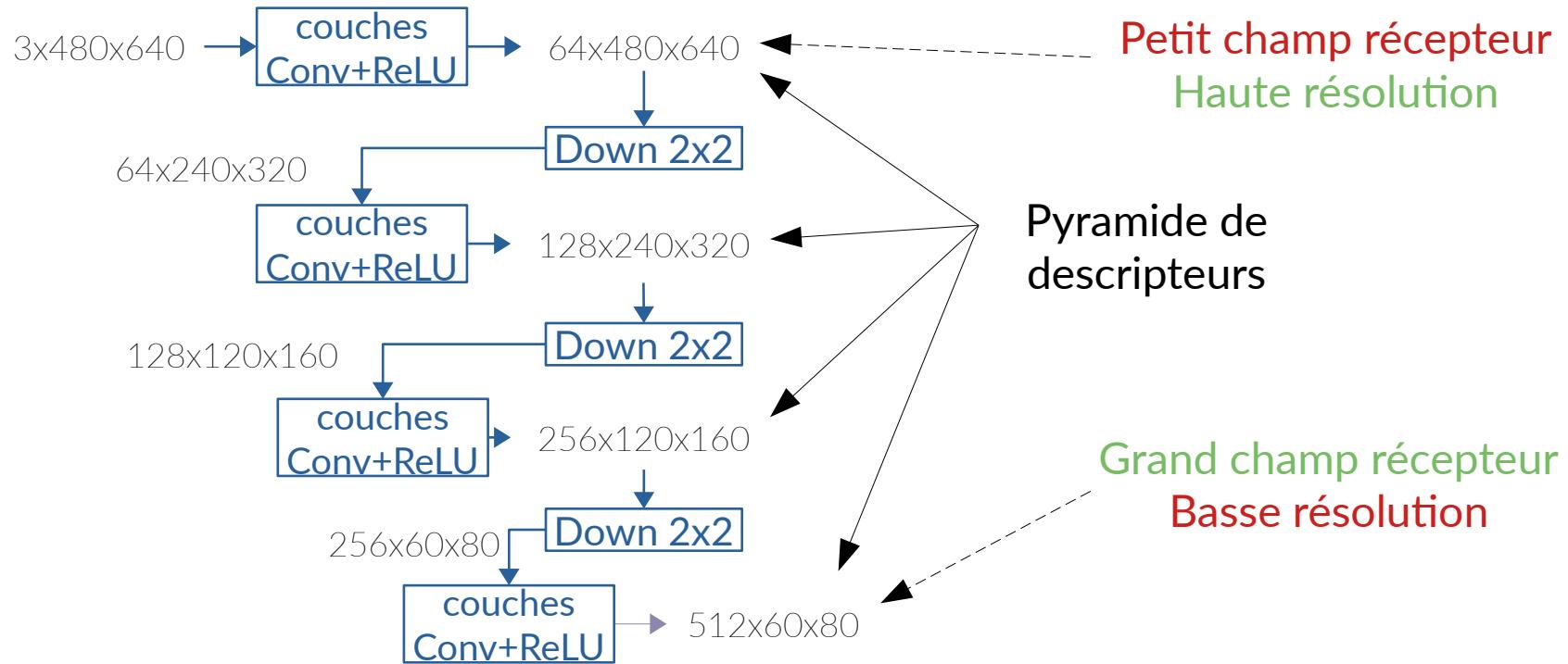
→ Croissance rapide du RF ... mais baisse de la résolution
→ Architecture U-Net

U-Net, également appelé « Feature Pyramid Network » (FPN)

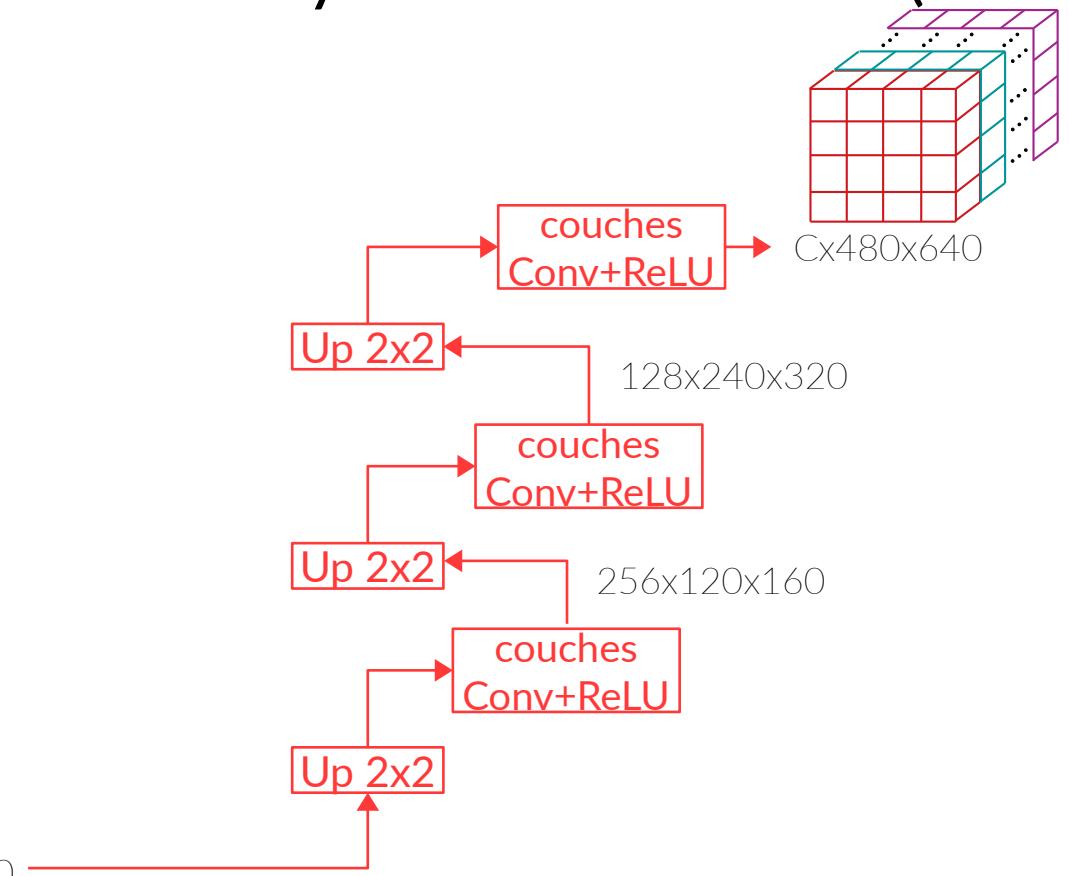
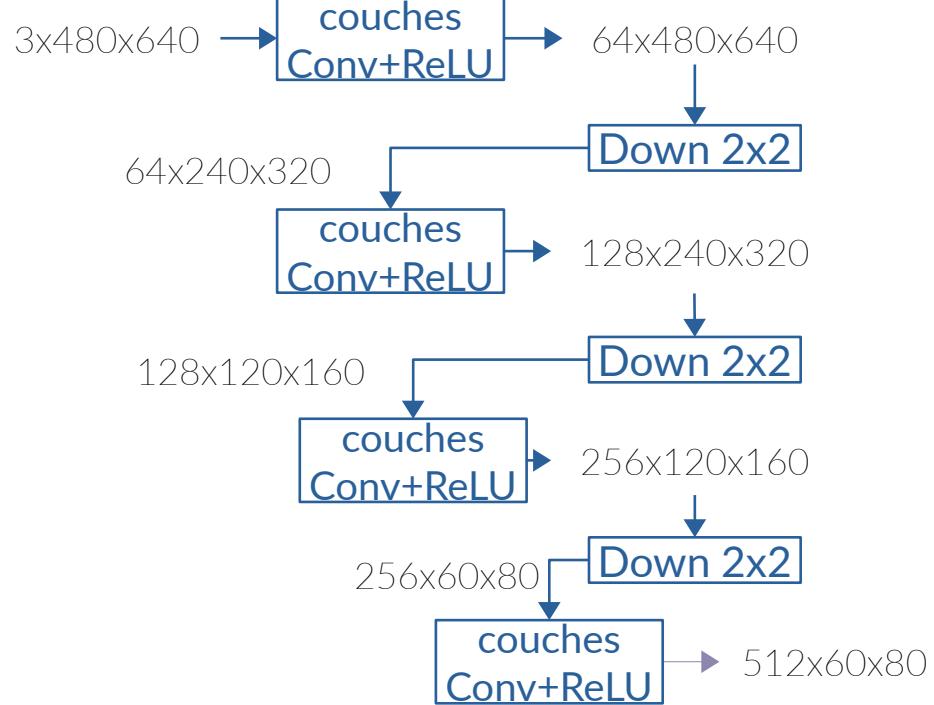


Pyramide de
descripteurs

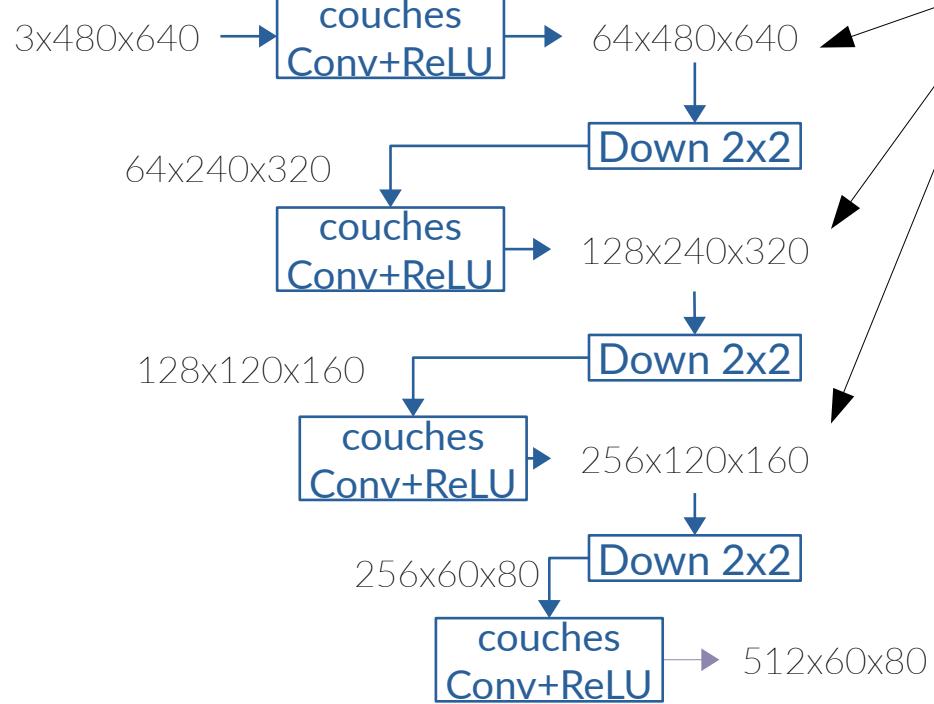
U-Net, également appelé « Feature Pyramid Network » (FPN)



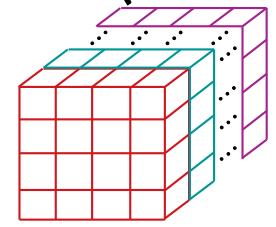
U-Net, également appelé « Feature Pyramid Network » (FPN)



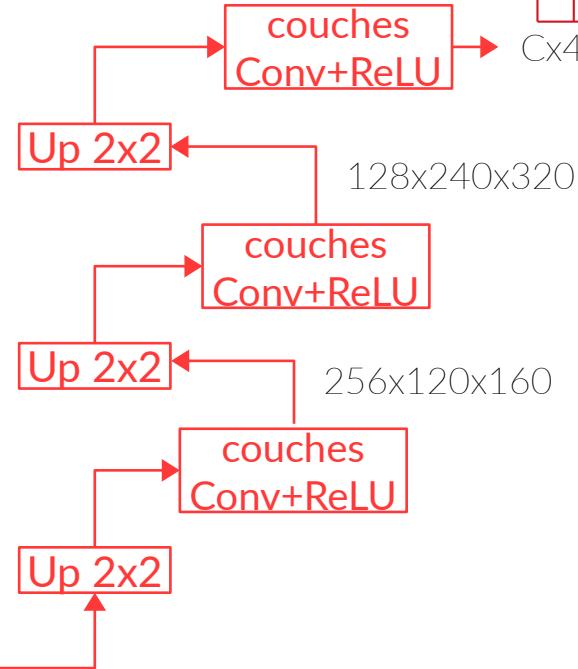
U-Net, également appelé « Feature Pyramid Network » (FPN)



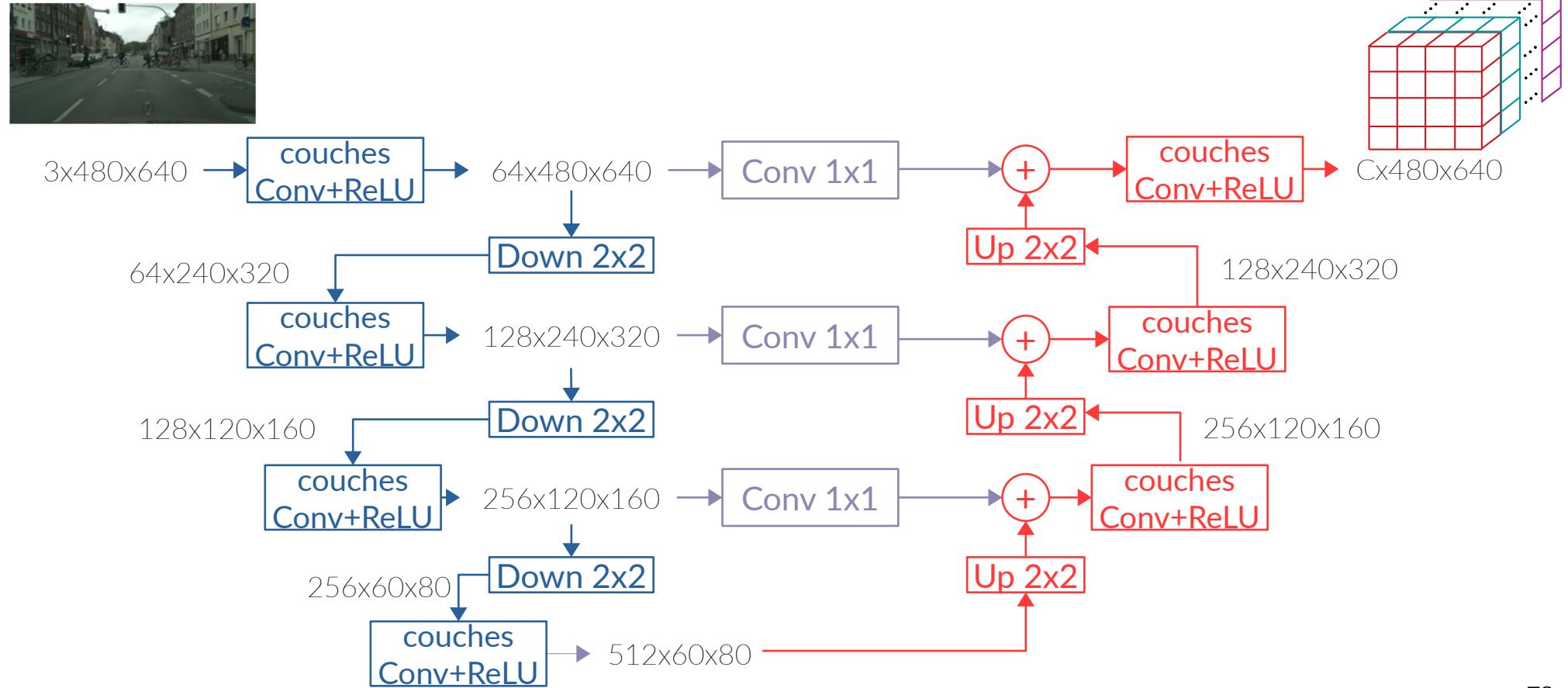
Les descripteurs « fins »
ne sont pas utilisés !



$C \times 480 \times 640$



U-Net, également appelé « Feature Pyramid Network » (FPN)



U-Net, également appelé « Feature Pyramid Network » (FPN)

