

Réseaux de neurones à convolution

Guillaume Bourmaud

PLAN

I. Couche de convolution

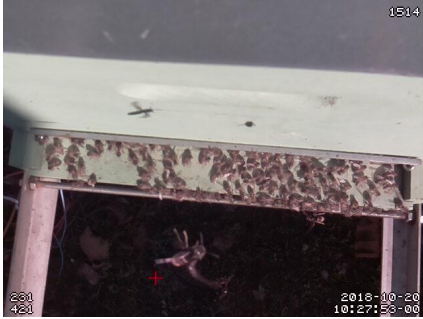
II. Réseaux de neurones à convolution

I) Couche de convolution

I)

Limites d'une transformation affine générale (FC)

H=480

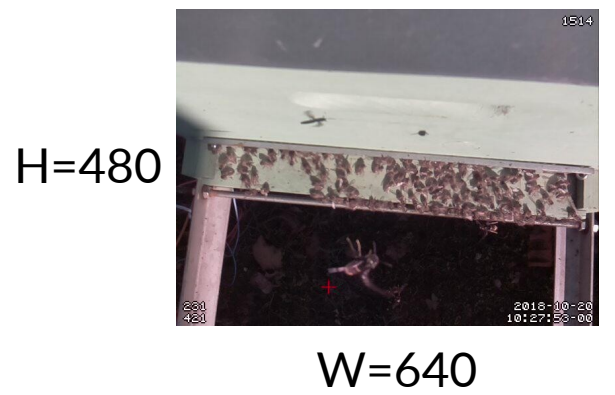


W=640

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

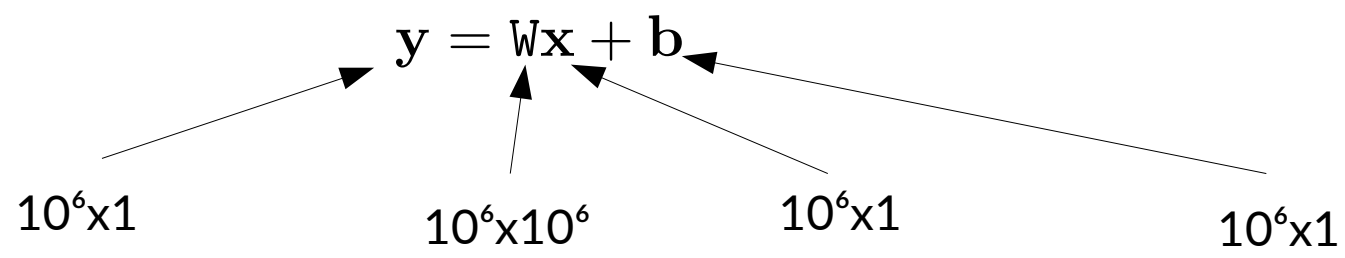
I)

Limites d'une transformation affine générale (FC)



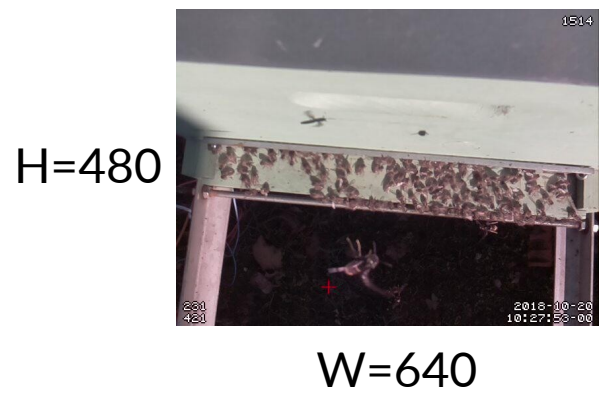
$x : 640 \times 480 \times 3 \approx 10^6$ éléments

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



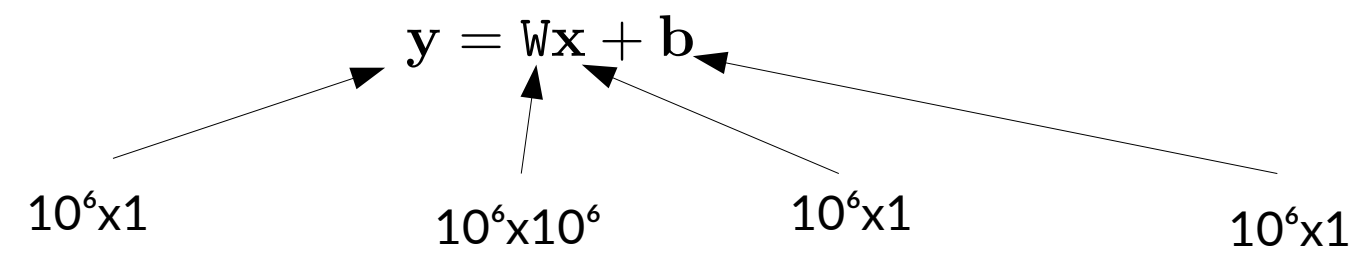
I)

Limites d'une transformation affine générale (FC)



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

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.

I)

Opération de « convolution » = transformation affine spécifique

“Fully Connected”

=

Transformation affine générale

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

I)

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

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

I)

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

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

Équivariance par translation
=
« Convolution »

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

I)

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

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

Équivariance par translation
=
« Convolution »

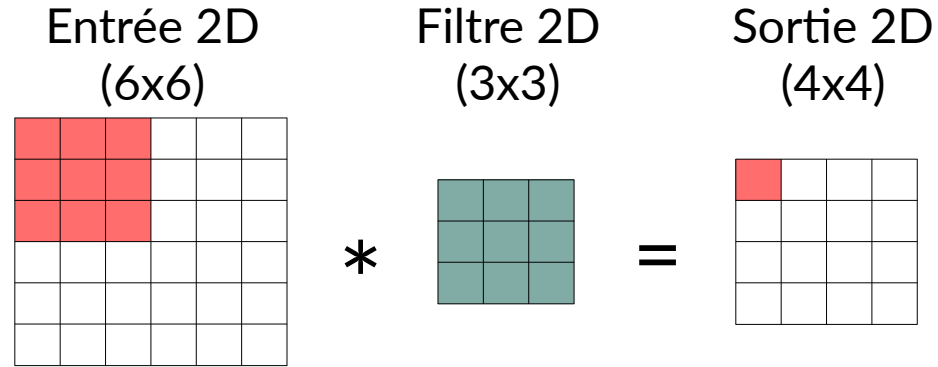
$$\begin{bmatrix} W_{11} & W_{12} & 0 & 0 \\ 0 & W_{11} & W_{12} & 0 \\ 0 & 0 & W_{11} & W_{12} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix}$$

Beaucoup moins de paramètres à stocker

Beaucoup moins de « multiplications+additions »

I)

Opération de « convolution » en 2D

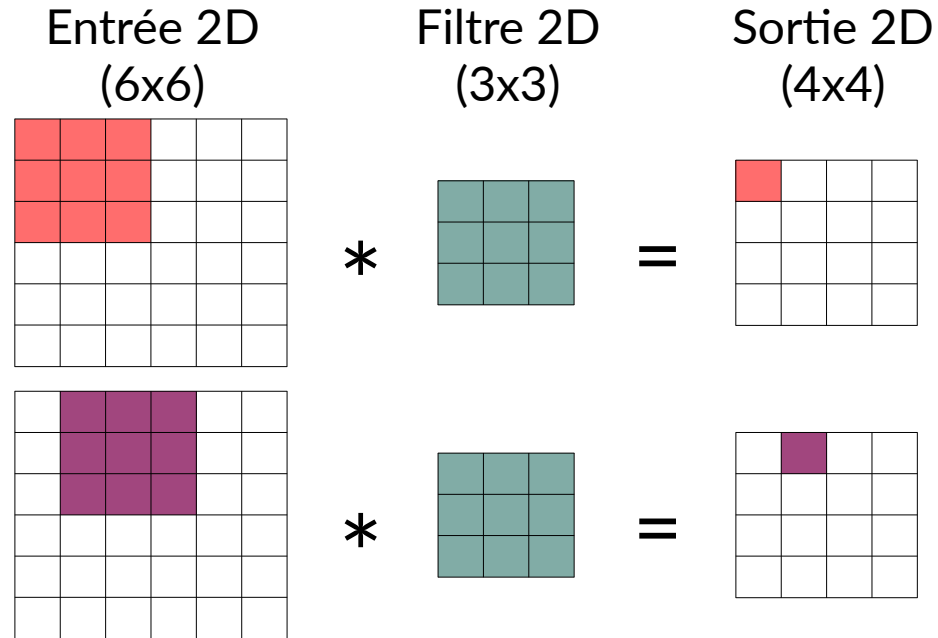


En fait, il s'agit d'une intercorrélacion

I)

Opération de « convolution » en 2D

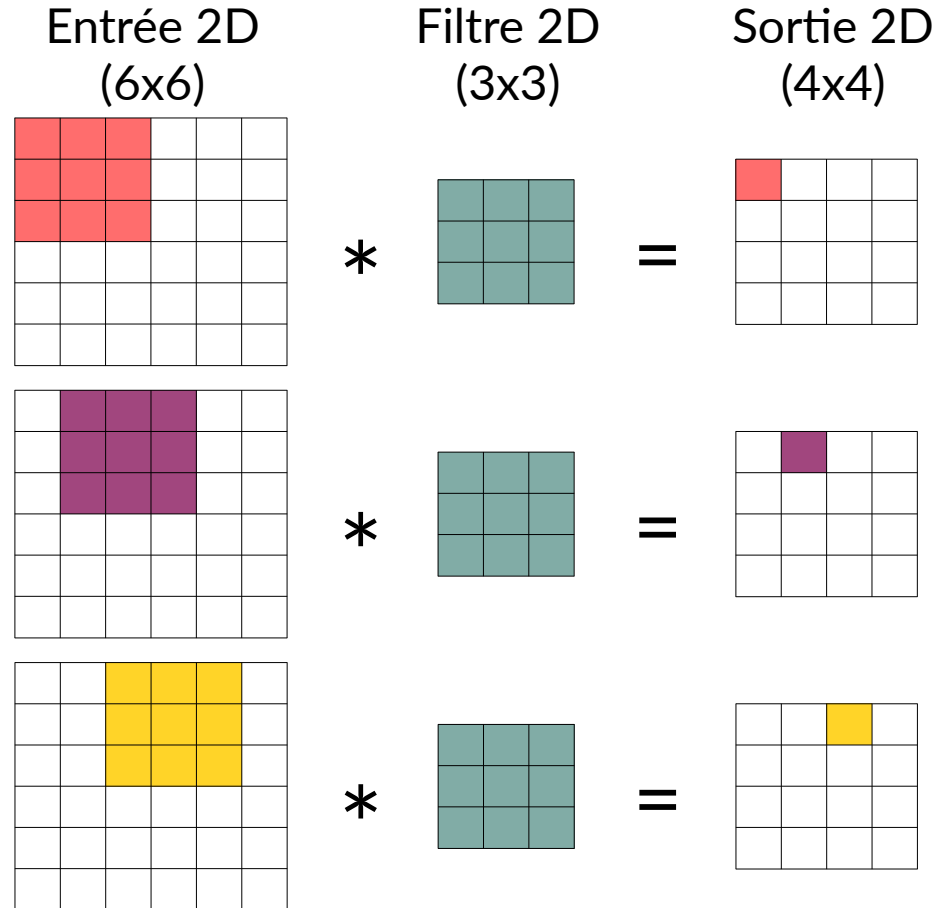
En fait, il s'agit d'une intercorrélation



I)

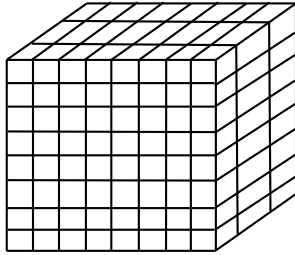
Opération de « convolution » en 2D

En fait, il s'agit d'une intercorrélation



1)

Couche de convolution 2D : un seul filtre



$X^{(0)}$

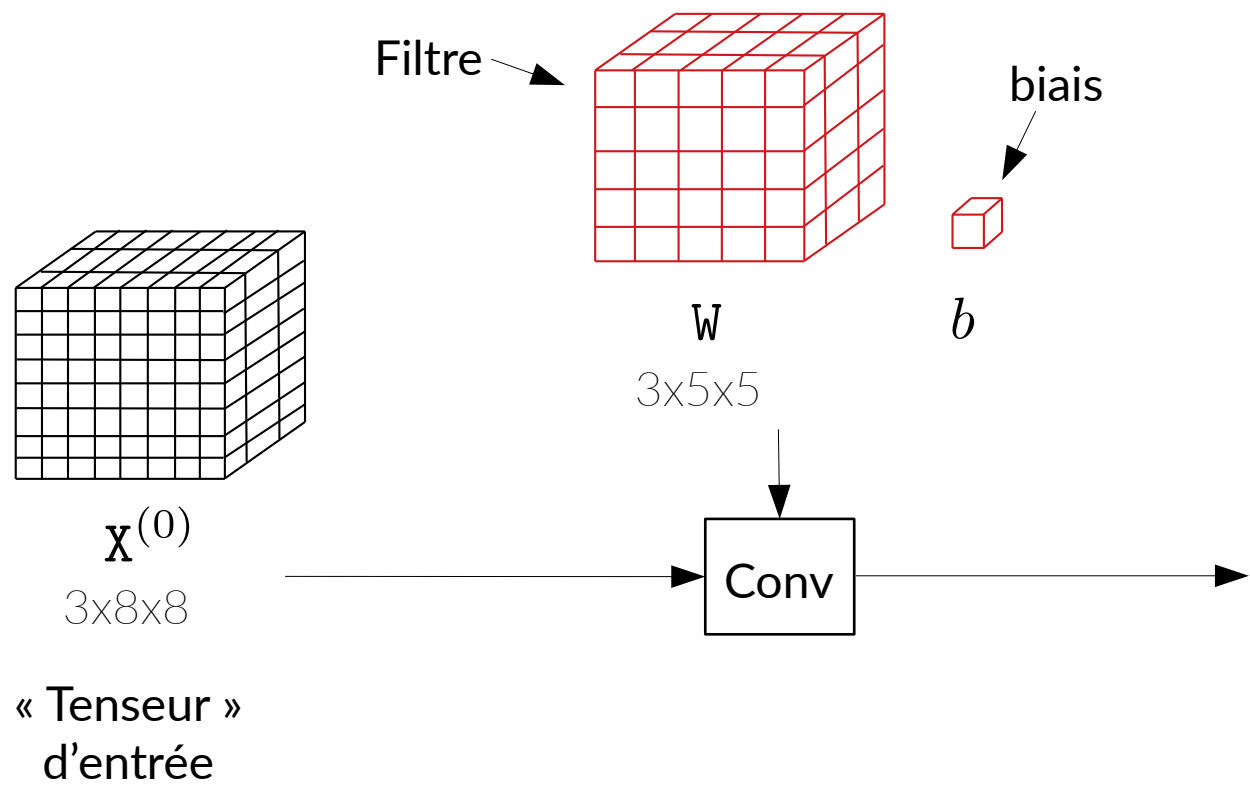
3x8x8

« Tenseur »
d'entrée

« Tenseur » = tableau multi-dimensionnel

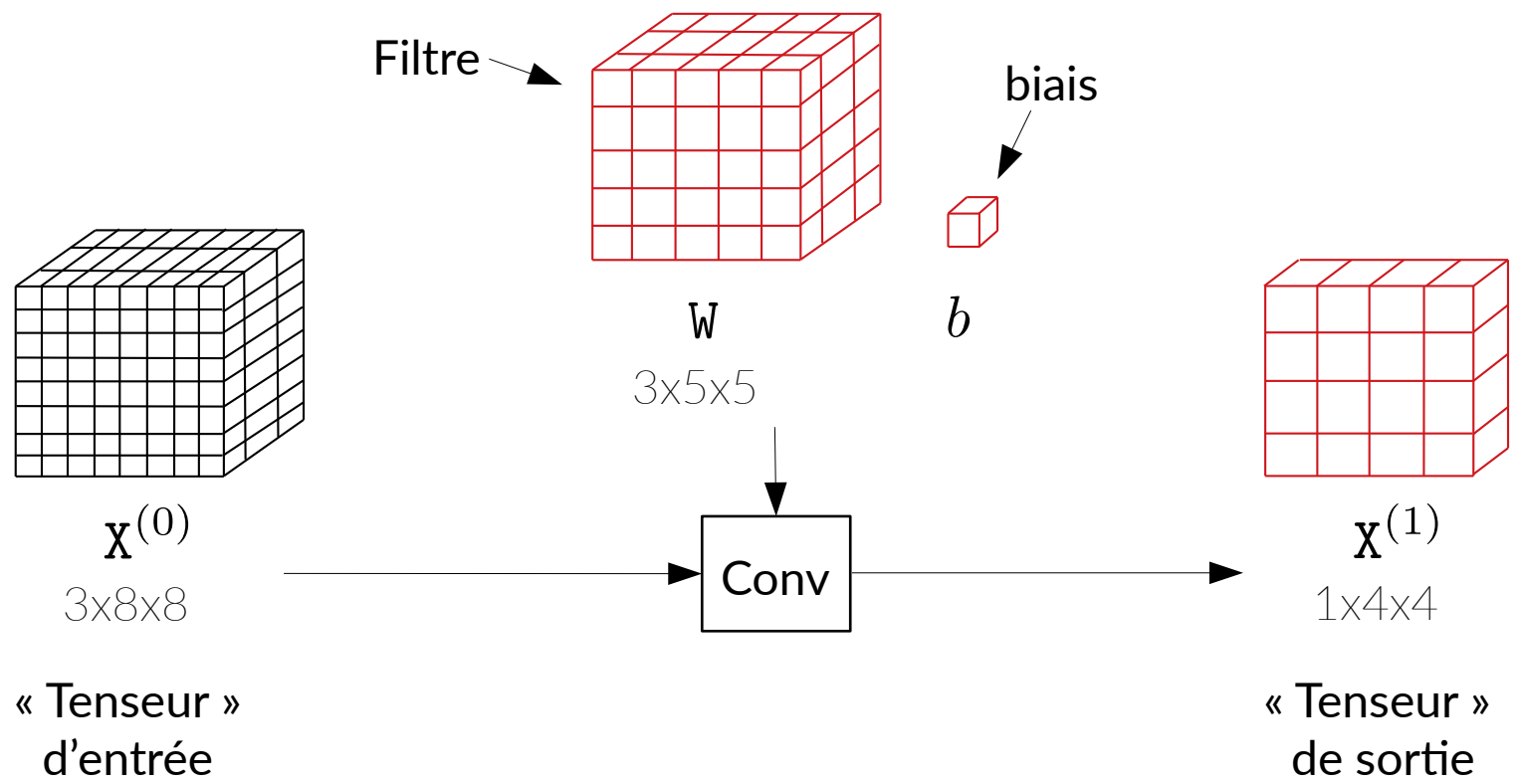
1)

Couche de convolution 2D : un seul filtre



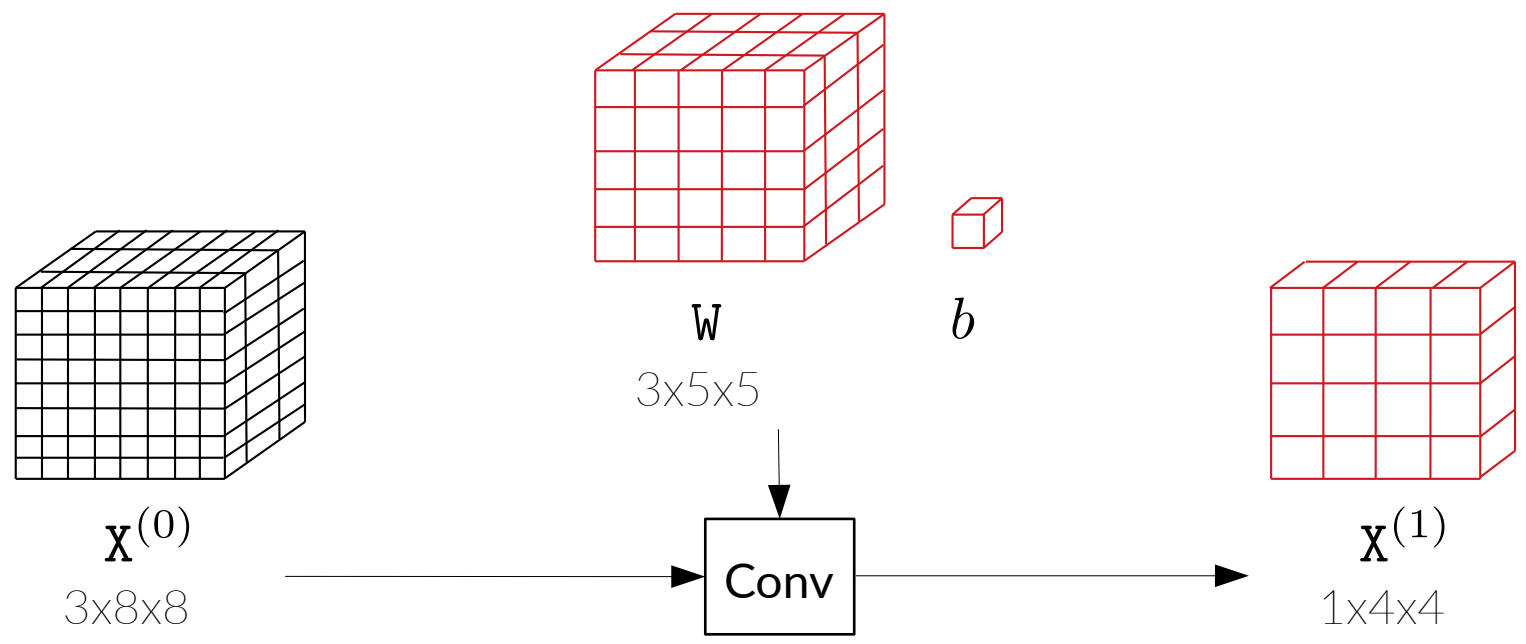
1)

Couche de convolution 2D : un seul filtre



1)

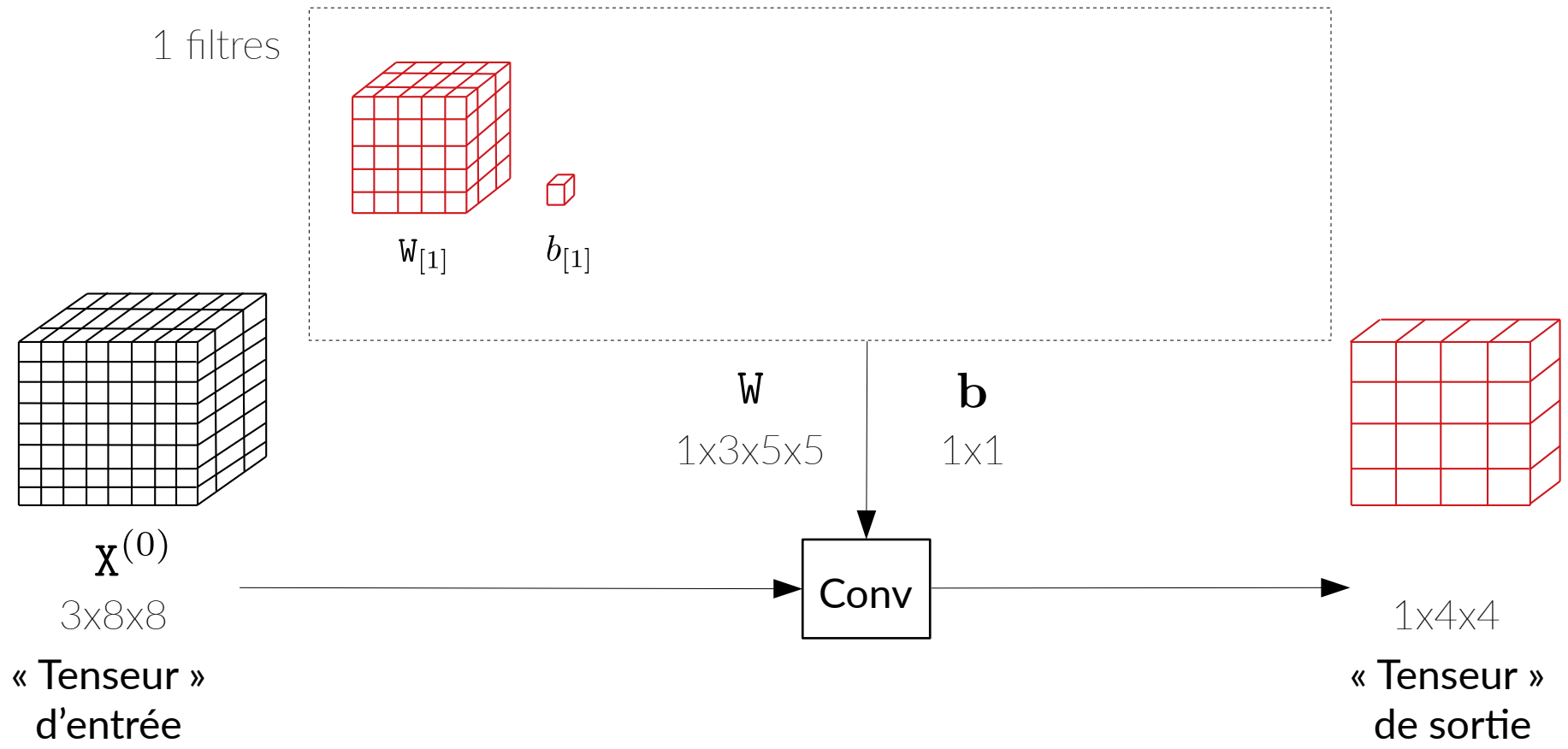
Couche de convolution 2D : un seul filtre



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

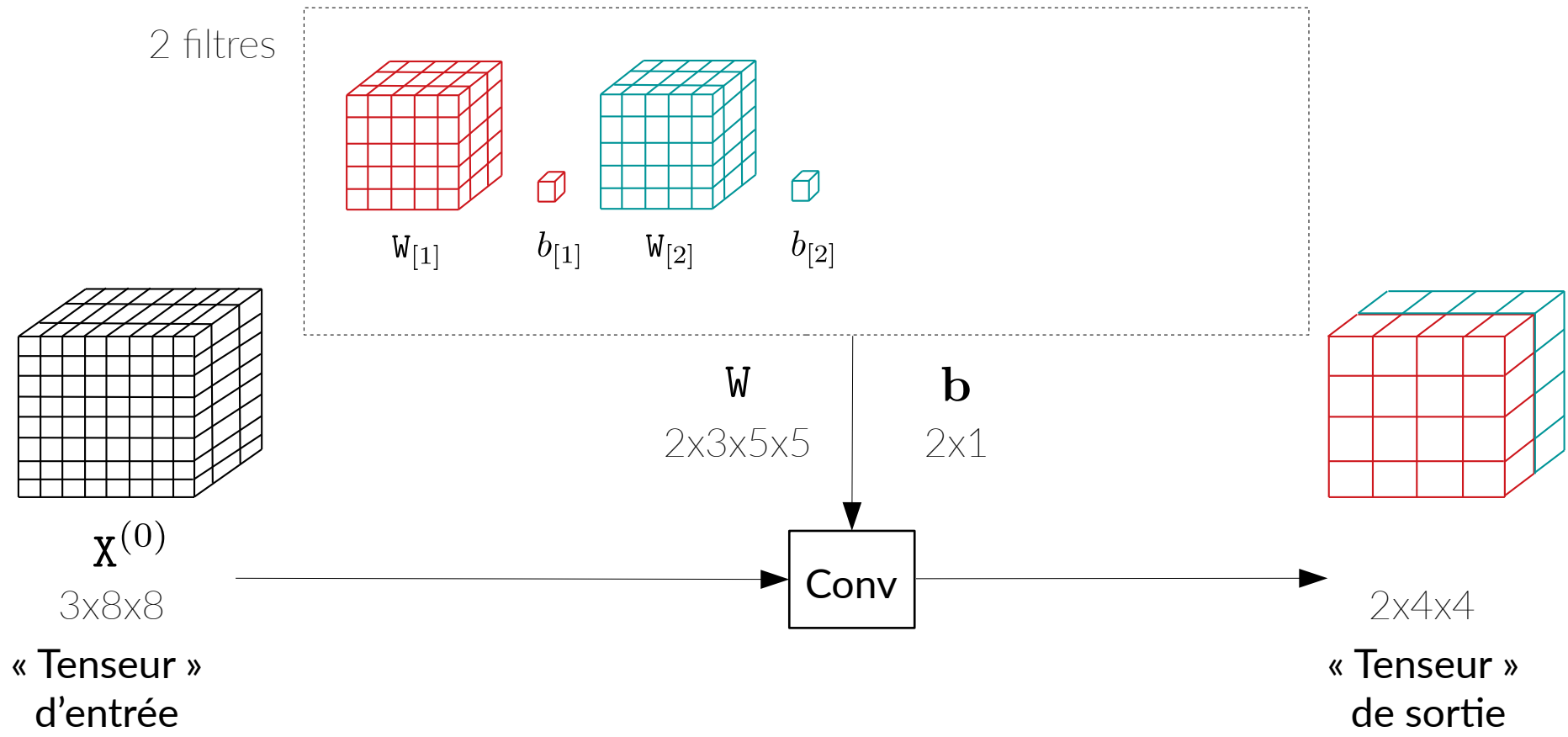
1)

Couche de convolution 2D : K filtres



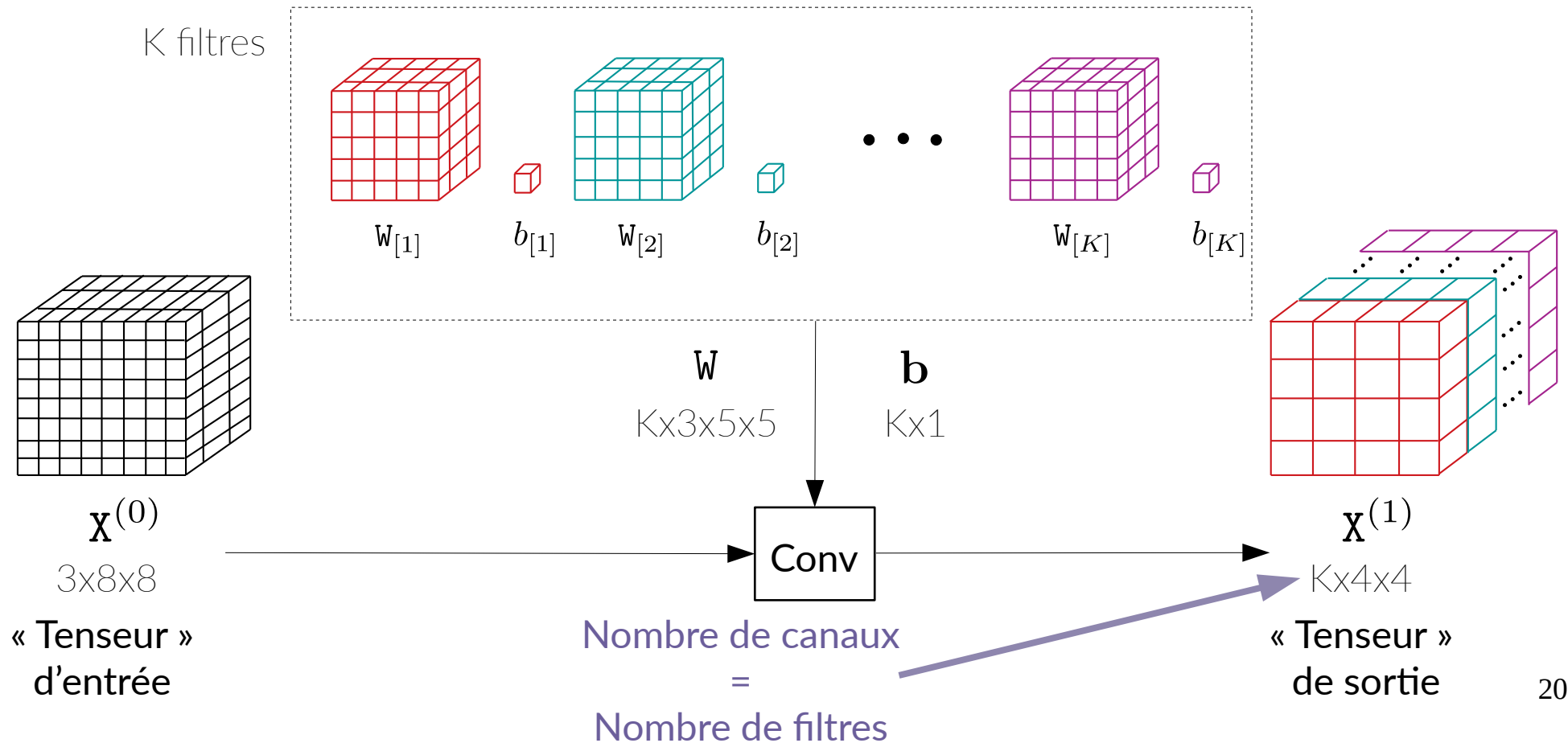
1)

Couche de convolution 2D : K filtres



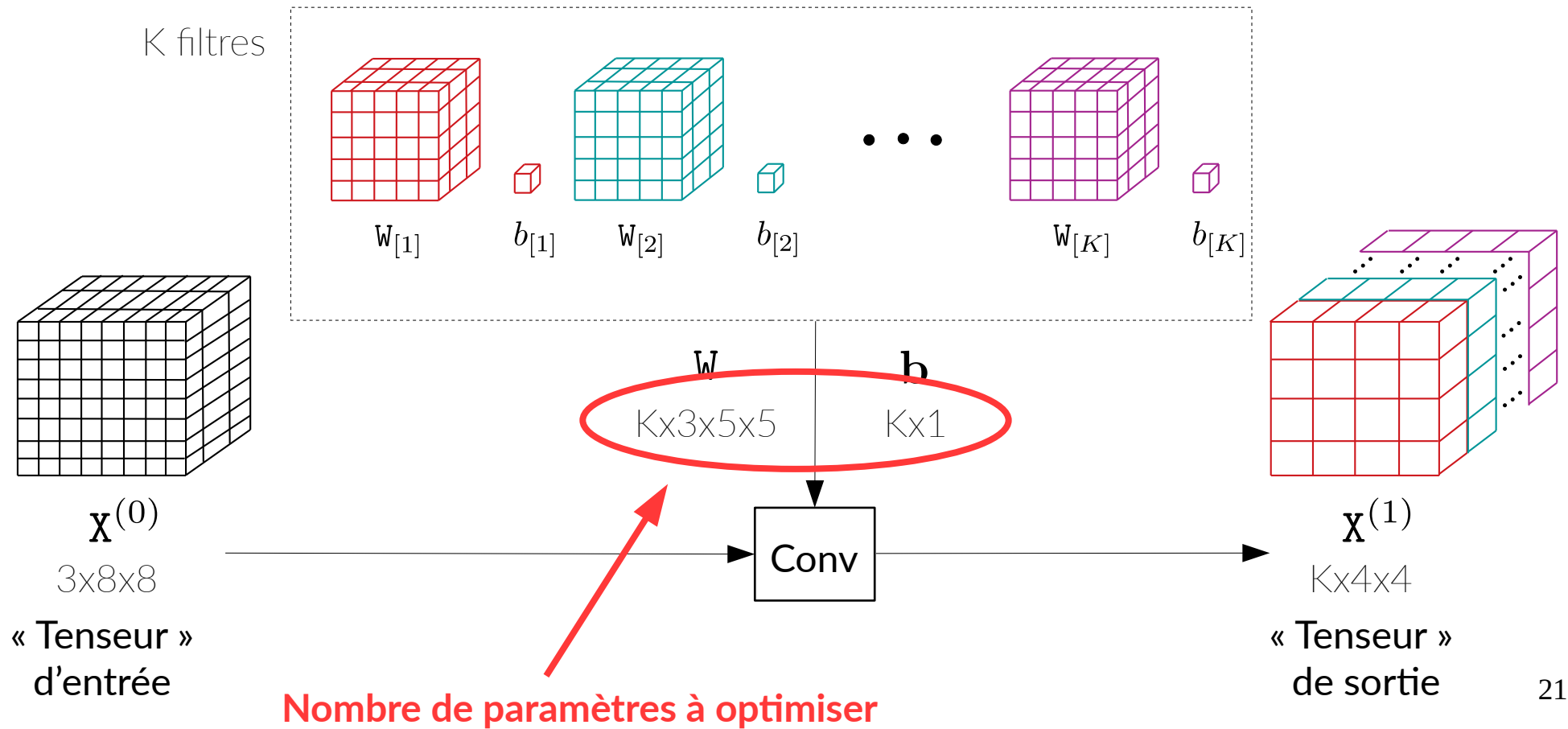
1)

Couche de convolution 2D : K filtres



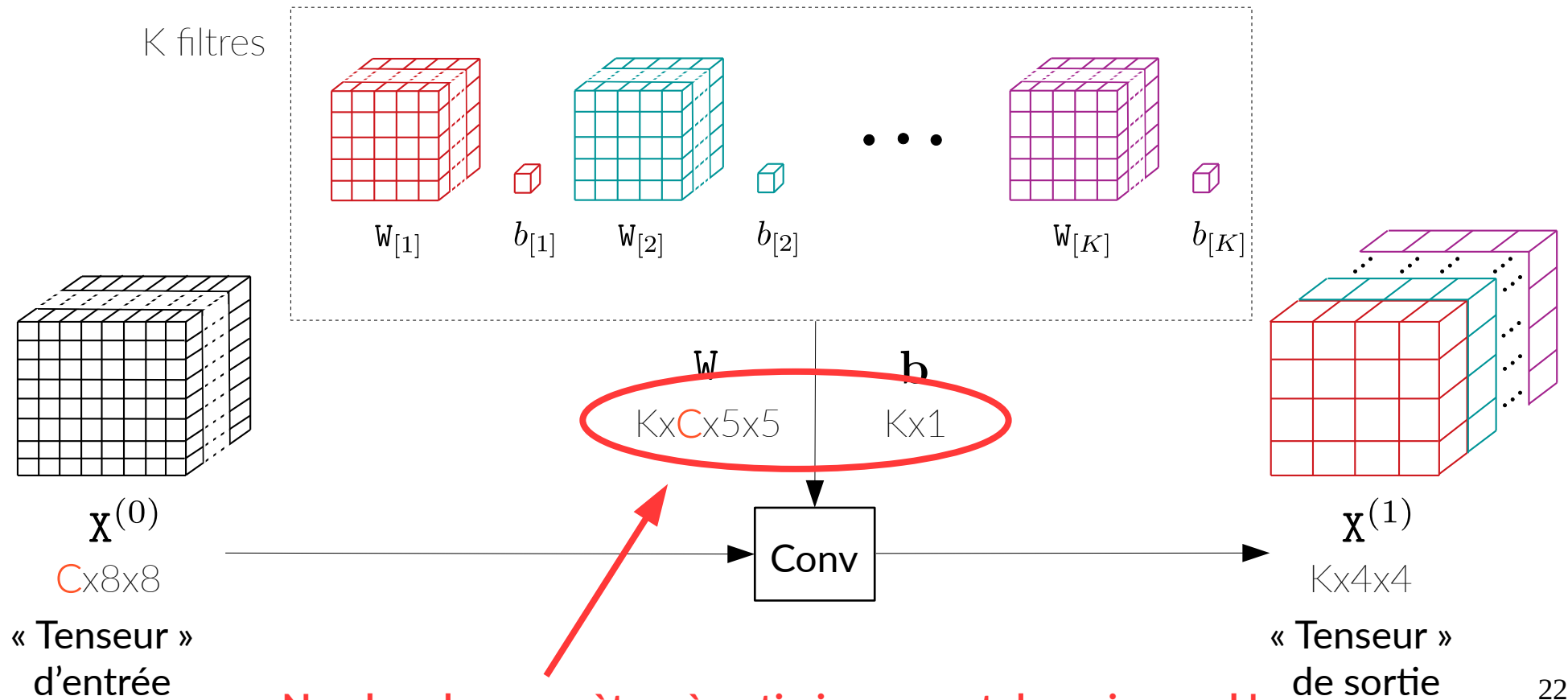
1)

Couche de convolution 2D : K filtres



1)

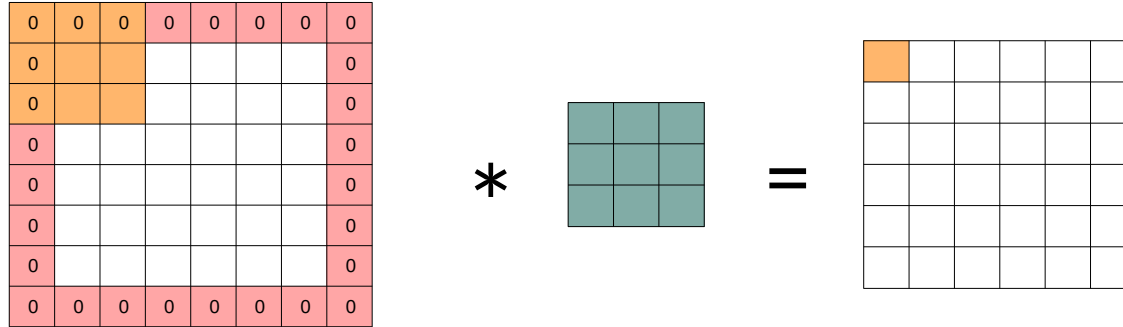
Couche de convolution 2D : K filtres



Nombre de paramètres à optimiser... peut devenir grand !

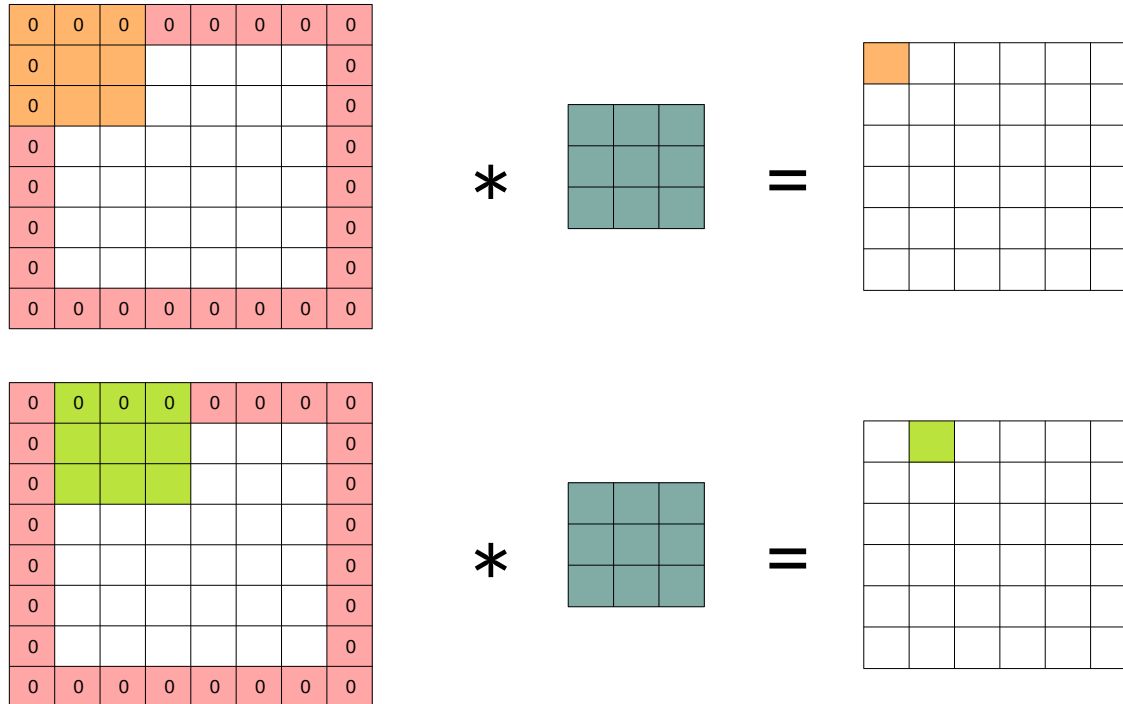
1)

« Zero padding »



1)

« Zero padding »

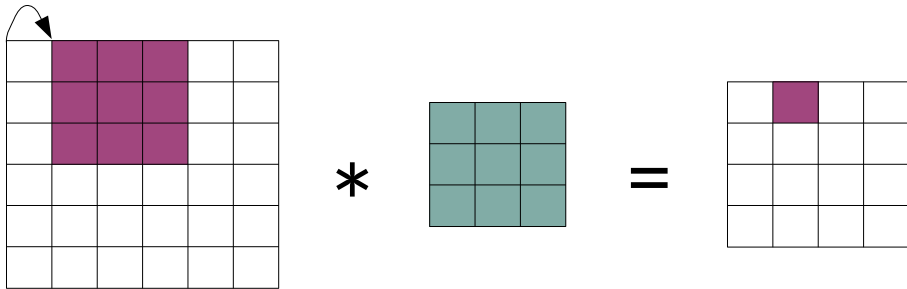
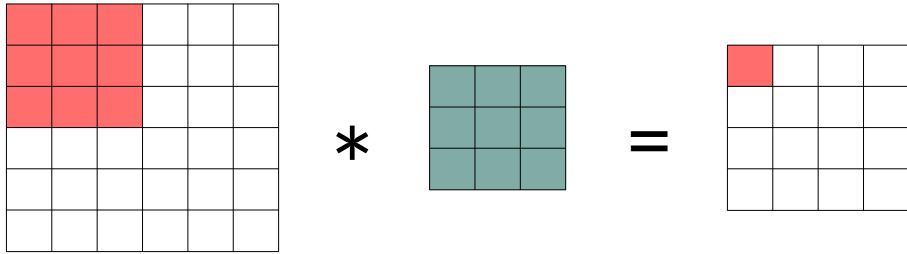


Permet de préserver la taille du tenseur d'entrée = ne pas avoir d'effet de bord

1)

« Stride »

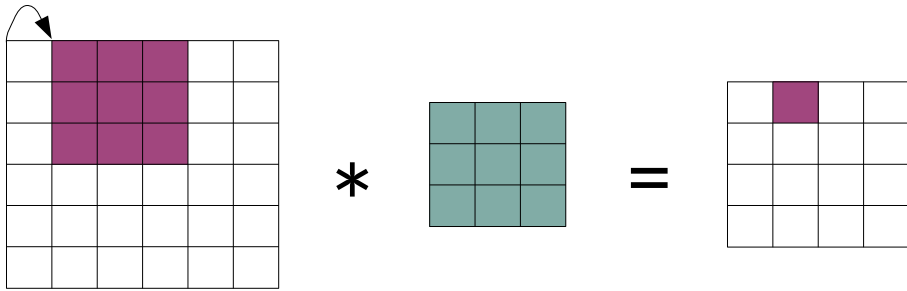
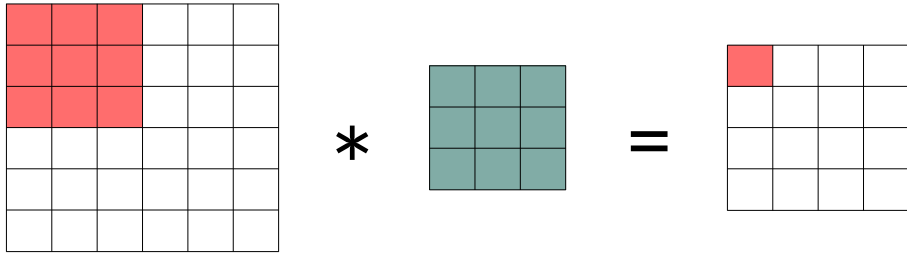
Exemple Stride = 1



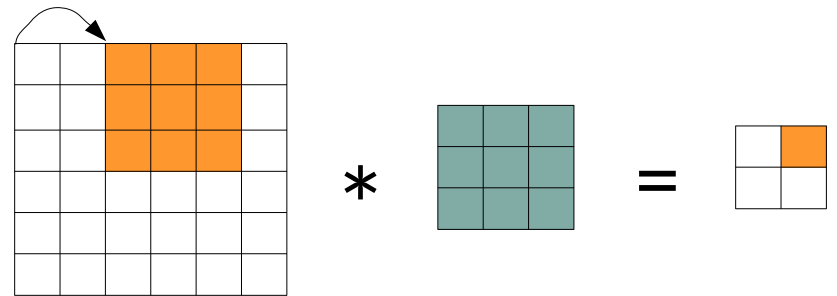
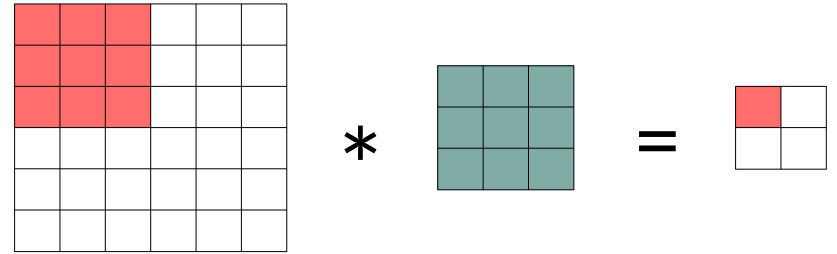
1)

« Stride »

Exemple Stride = 1



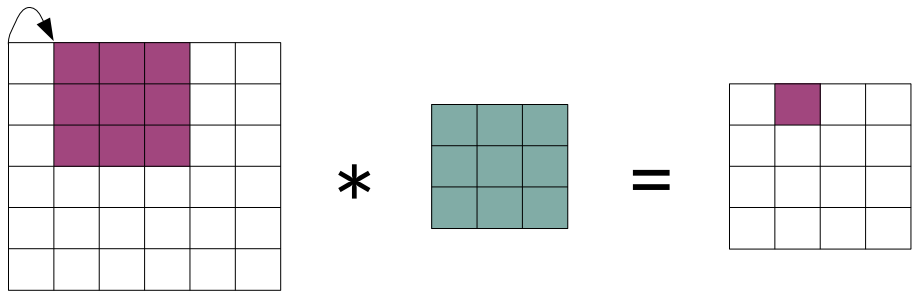
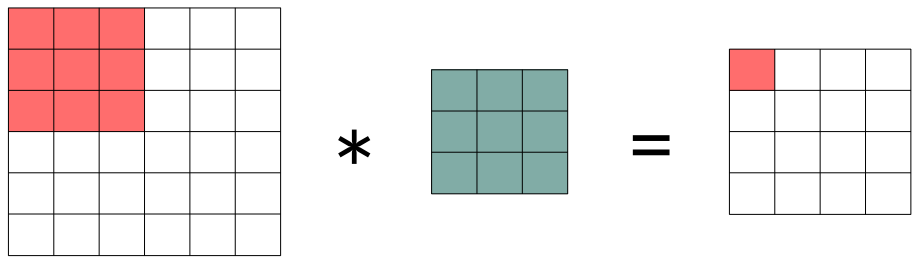
Exemple Stride = 2



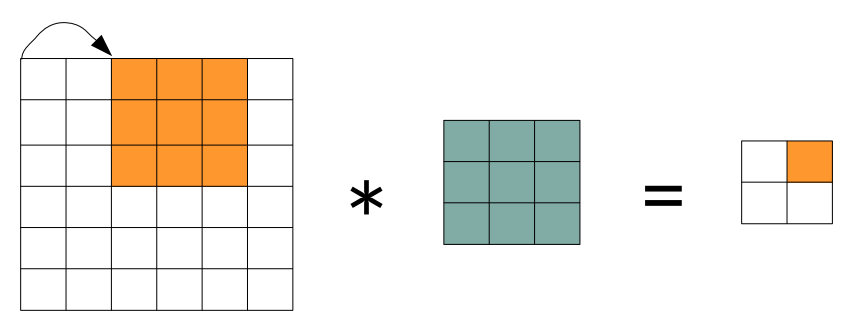
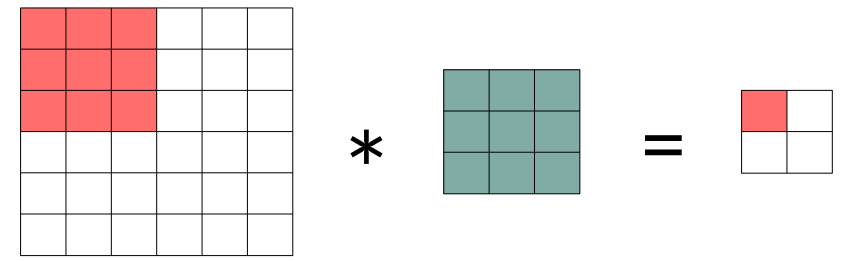
I)

« Stride »

Exemple Stride = 1



Exemple Stride = 2



Réduit la résolution de la sortie par « stride »

I)

Principaux hyperparamètres d'une couche de convolution

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

I)

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

I)

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
 - 1x1 → padding = 0, 3x3 → padding = 1, 5x5 → padding = 2

I)

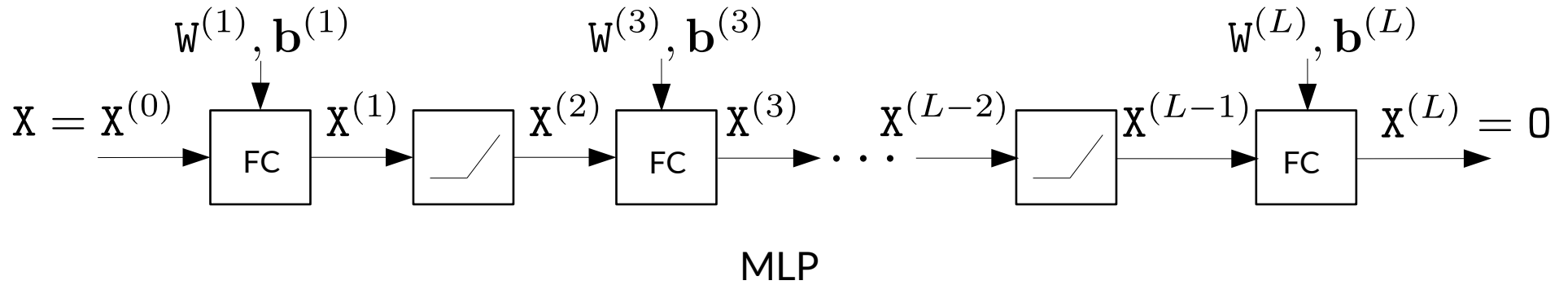
Principaux hyperparamètres d'une couche de convolution

- 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
 - 1x1 → padding = 0, 3x3 → padding = 1, 5x5 → padding = 2
- Stride
 - = 1 si volonté de préserver la résolution de l'entrée
 - = 2 si volonté de réduire la résolution de l'entrée

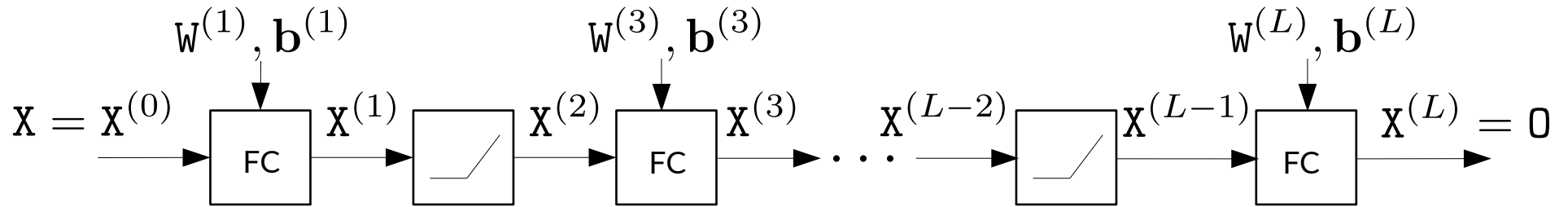
II) Réseau de neurones à convolution

II)

Réseau de neurones à convolution (CNN)



Réseau de neurones à convolution (CNN)



MLP

- FC (transformations affines générales)

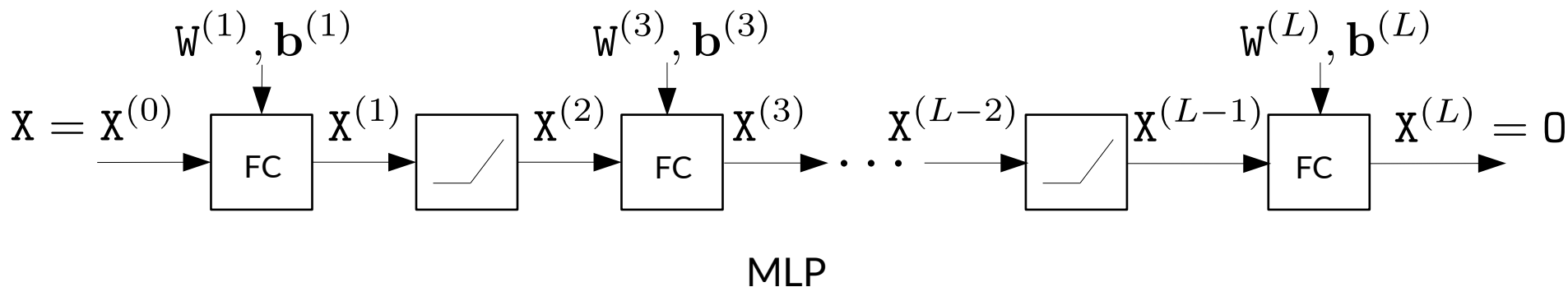
+ Conv (transformations affines spécifiques)

=

CNN

II)

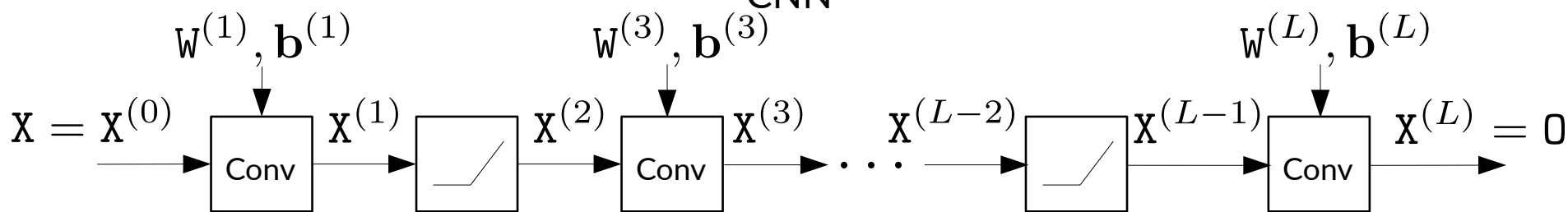
Réseau de neurones à convolution (CNN)



- FC (transformations affines générales)

+ Conv (transformations affines spécifiques)

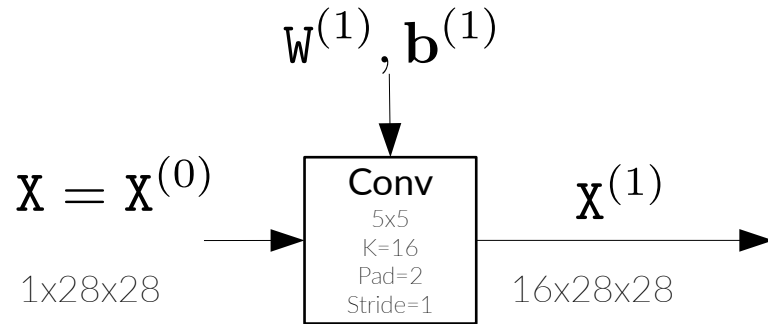
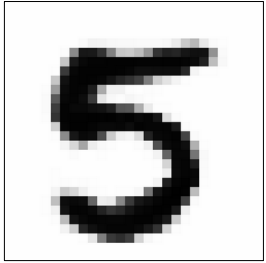
= CNN



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

II)

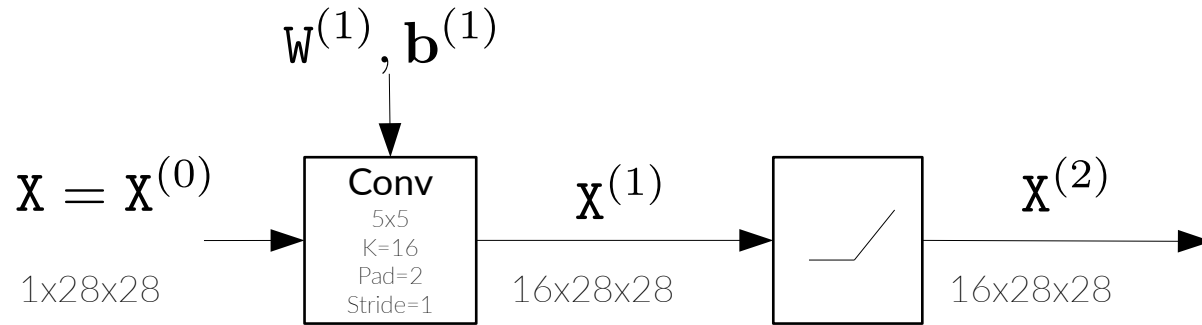
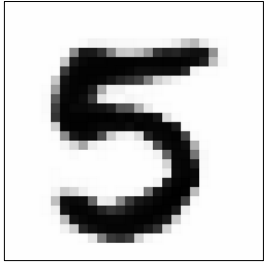
Exemple d'architecture de CNN pour MNIST



- "0"
- "1"
- "2"
- "3"
- "4"
- "5"
- "6"
- "7"
- "8"
- "9"

II)

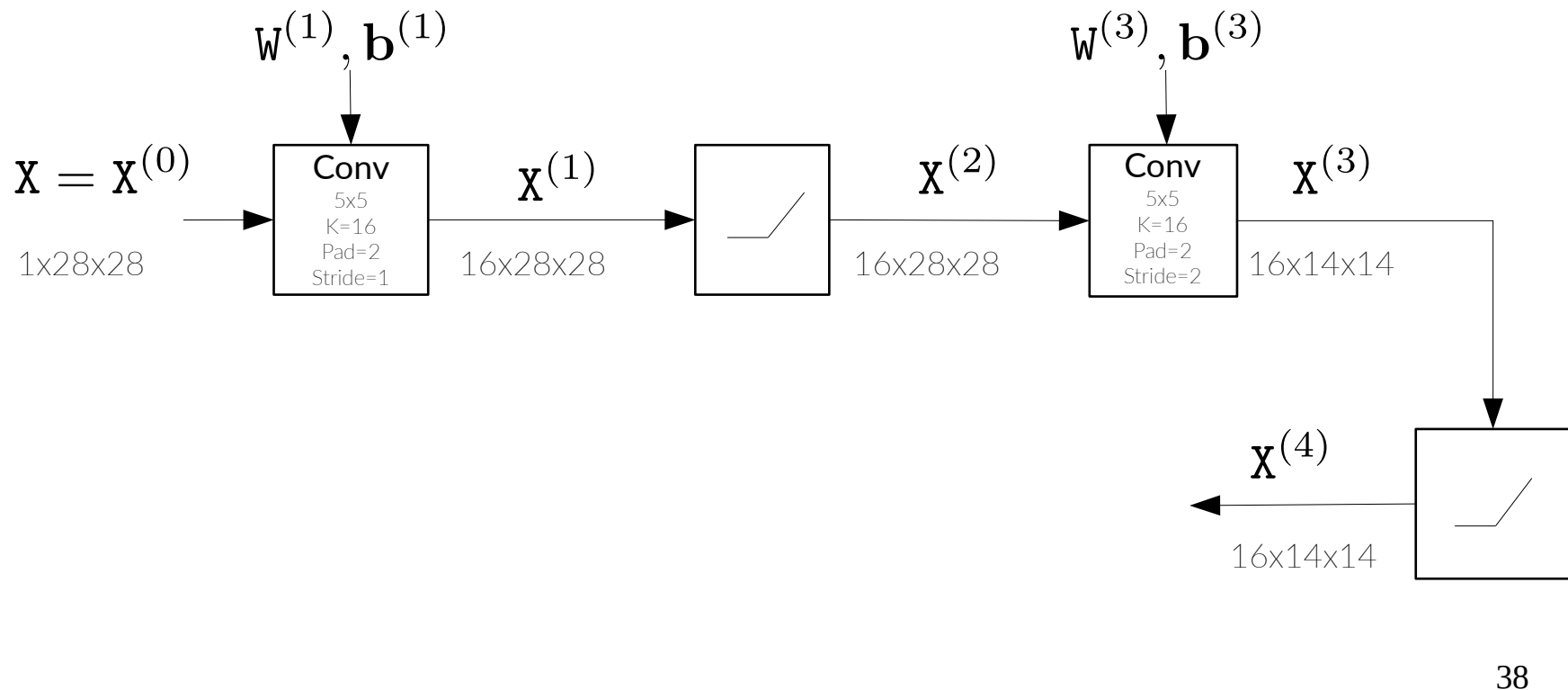
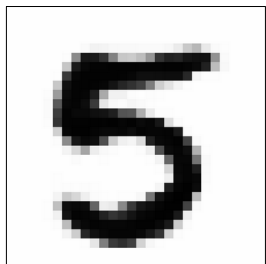
Exemple d'architecture de CNN pour MNIST



- "0"
- "1"
- "2"
- "3"
- "4"
- "5"
- "6"
- "7"
- "8"
- "9"

II)

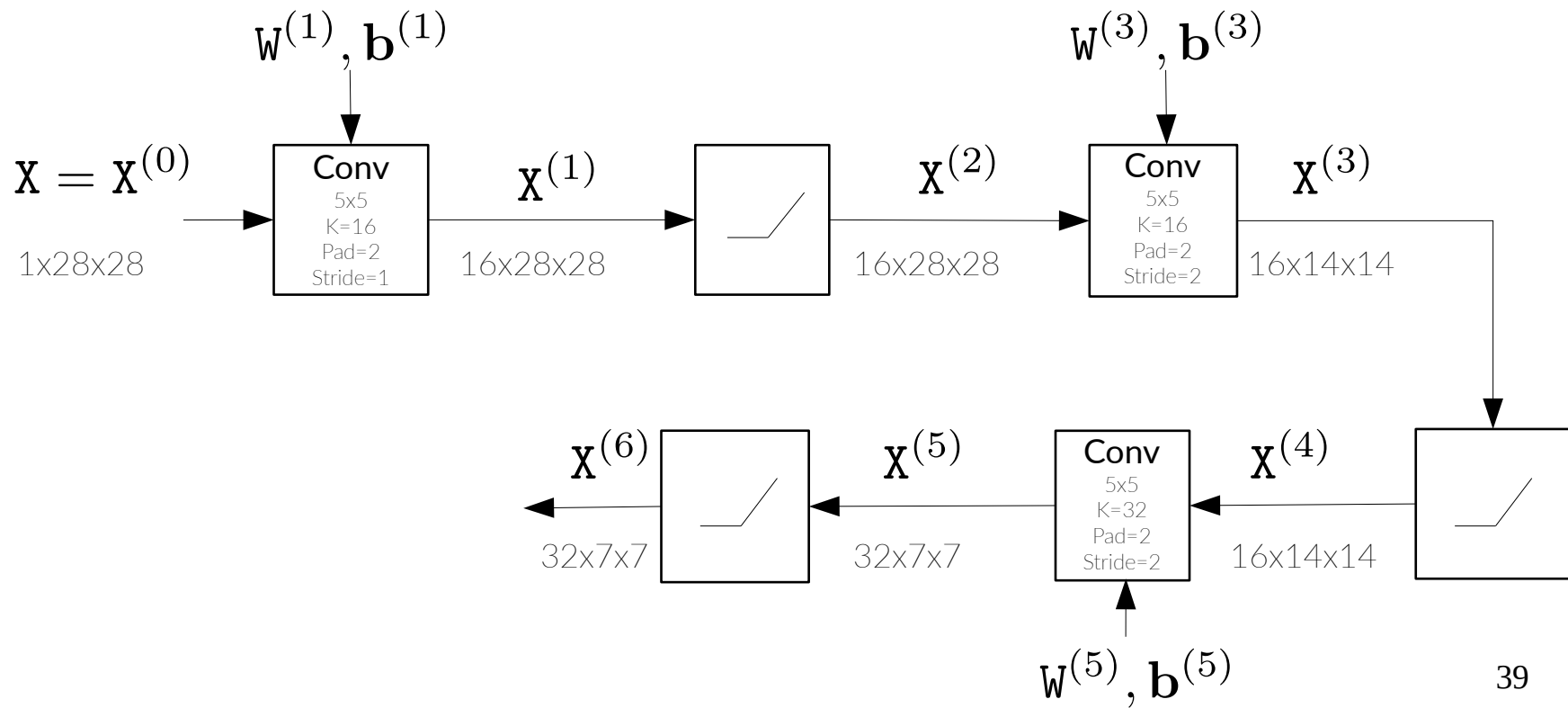
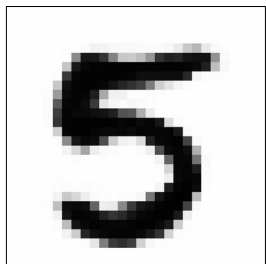
Exemple d'architecture de CNN pour MNIST



- "0"
- "1"
- "2"
- "3"
- "4"
- "5"
- "6"
- "7"
- "8"
- "9"

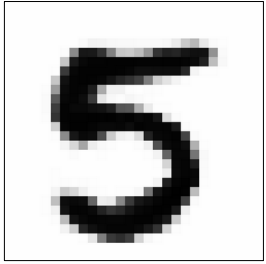
II)

Exemple d'architecture de CNN pour MNIST

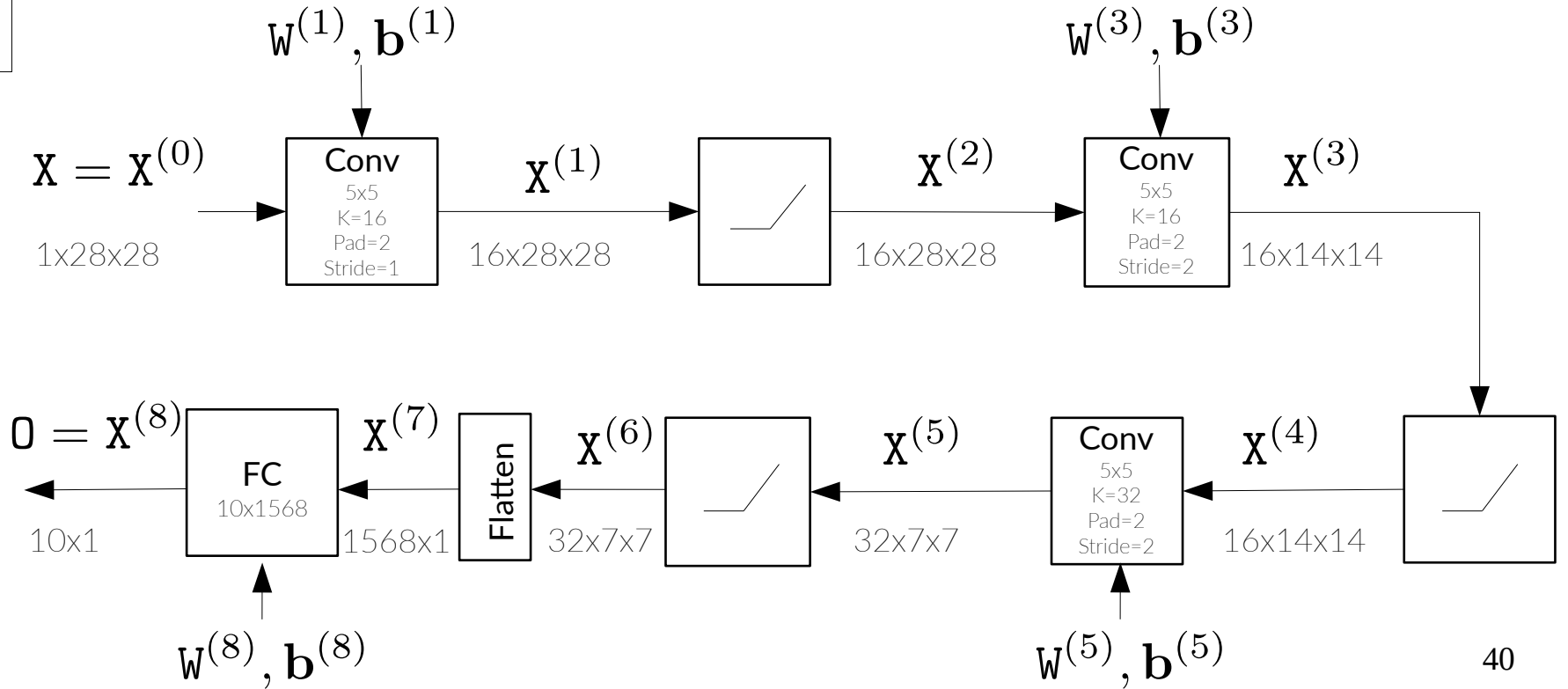


II)

Exemple d'architecture de CNN pour MNIST

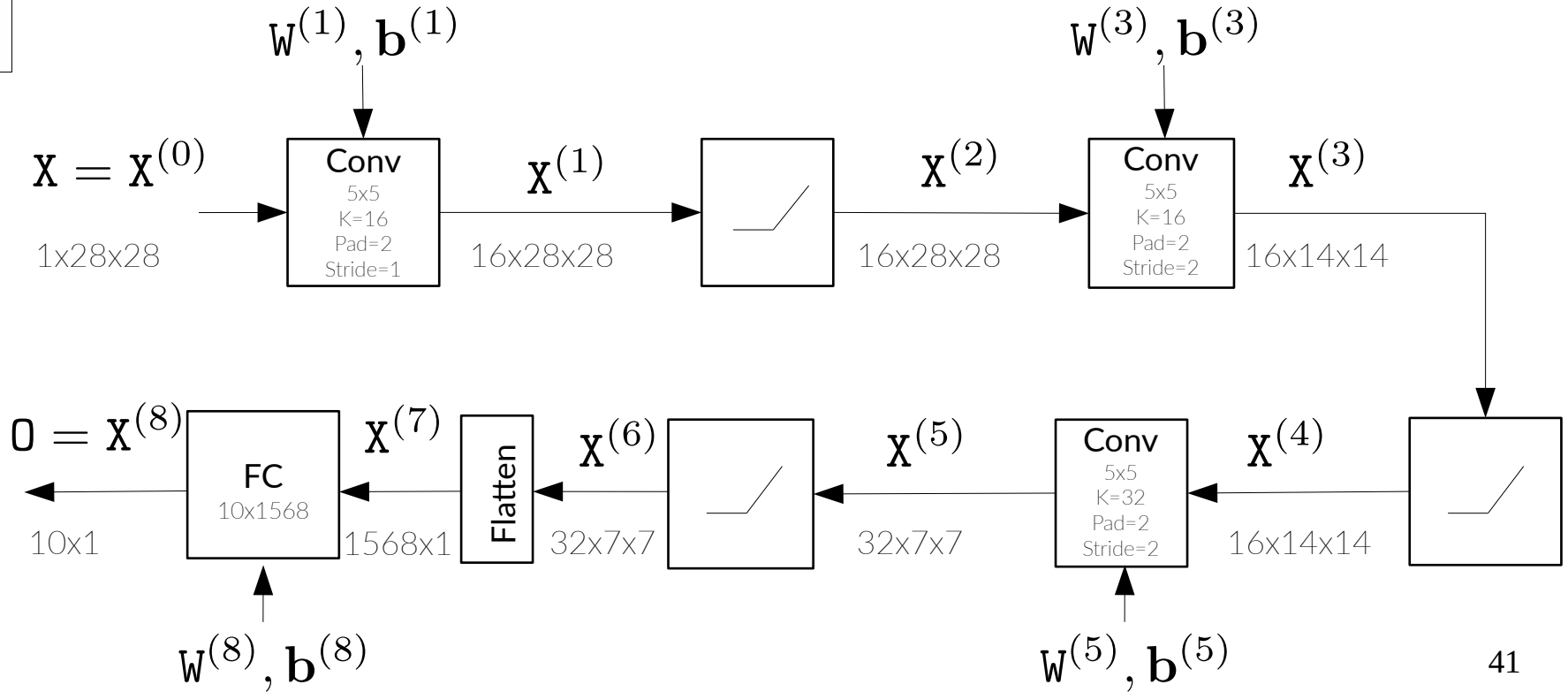
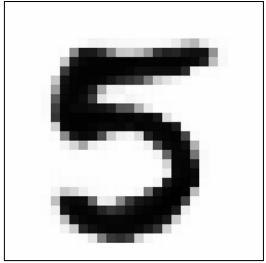


- "0"
- "1"
- "2"
- "3"
- "4"
- "5"
- "6"
- "7"
- "8"
- "9"



II)

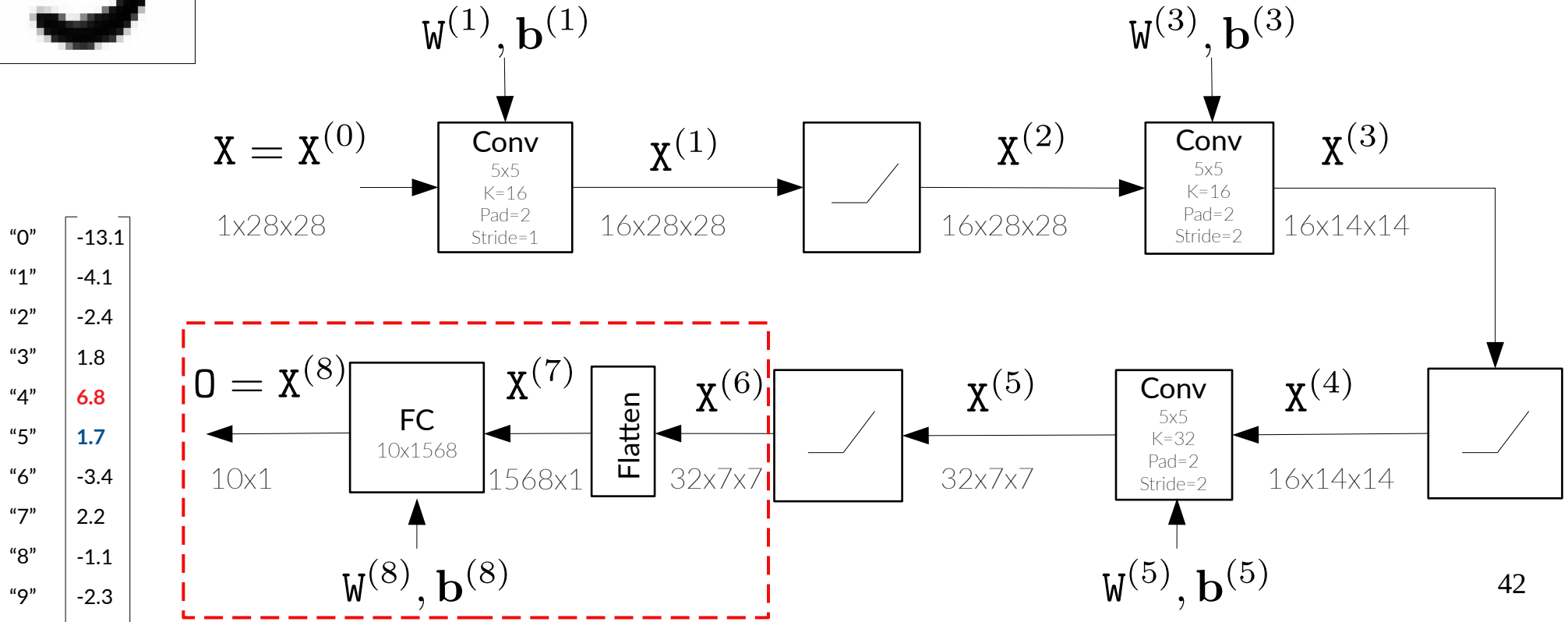
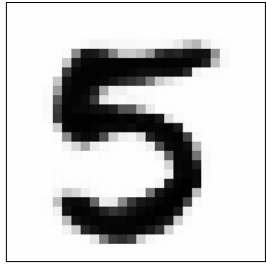
Exemple d'architecture de CNN pour MNIST



| | |
|-----|-------|
| "0" | -13.1 |
| "1" | -4.1 |
| "2" | -2.4 |
| "3" | 1.8 |
| "4" | 6.8 |
| "5" | 1.7 |
| "6" | -3.4 |
| "7" | 2.2 |
| "8" | -1.1 |
| "9" | -2.3 |

II)

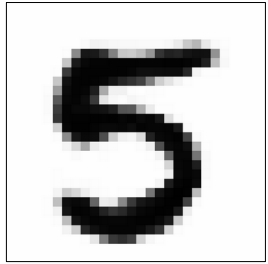
Exemple d'architecture de CNN pour MNIST



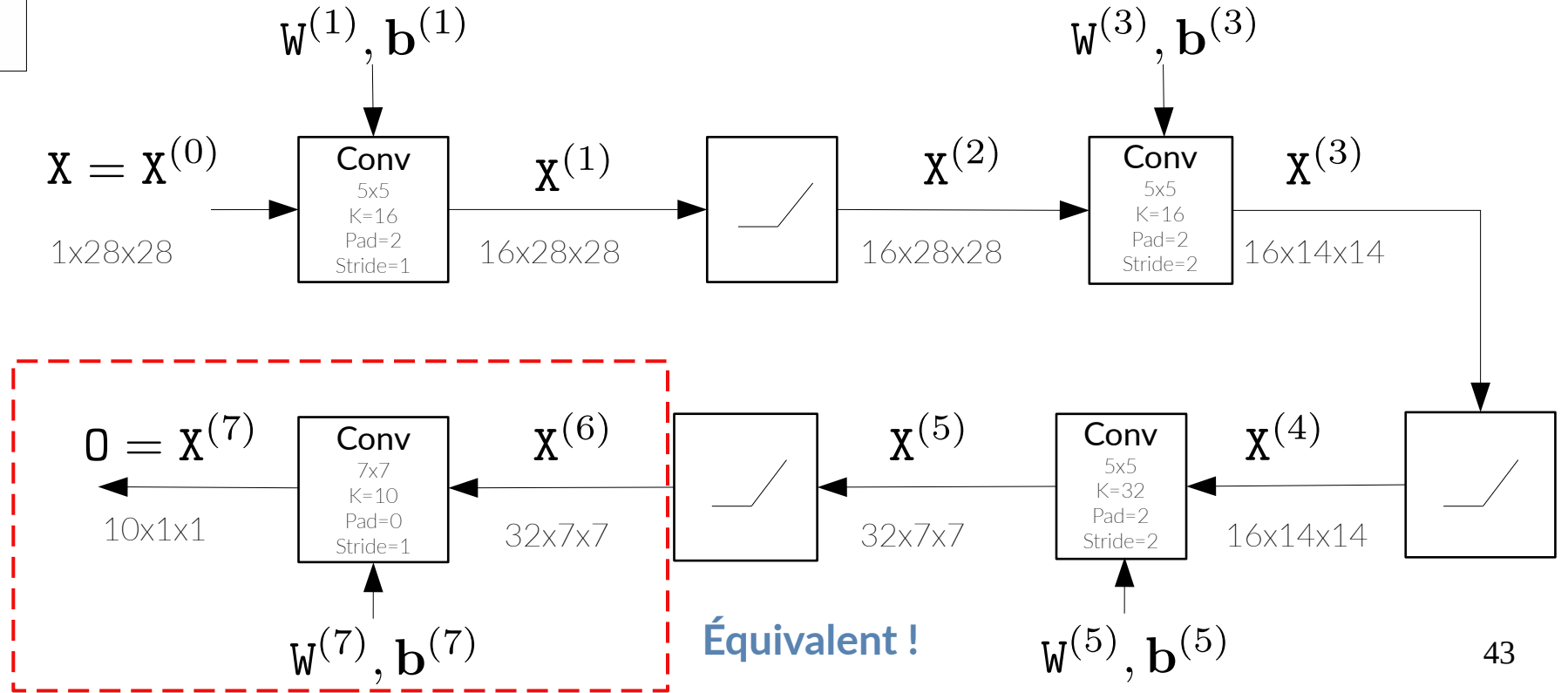
| | |
|-----|-------|
| "0" | -13.1 |
| "1" | -4.1 |
| "2" | -2.4 |
| "3" | 1.8 |
| "4" | 6.8 |
| "5" | 1.7 |
| "6" | -3.4 |
| "7" | 2.2 |
| "8" | -1.1 |
| "9" | -2.3 |

II)

Exemple d'architecture de CNN pour MNIST

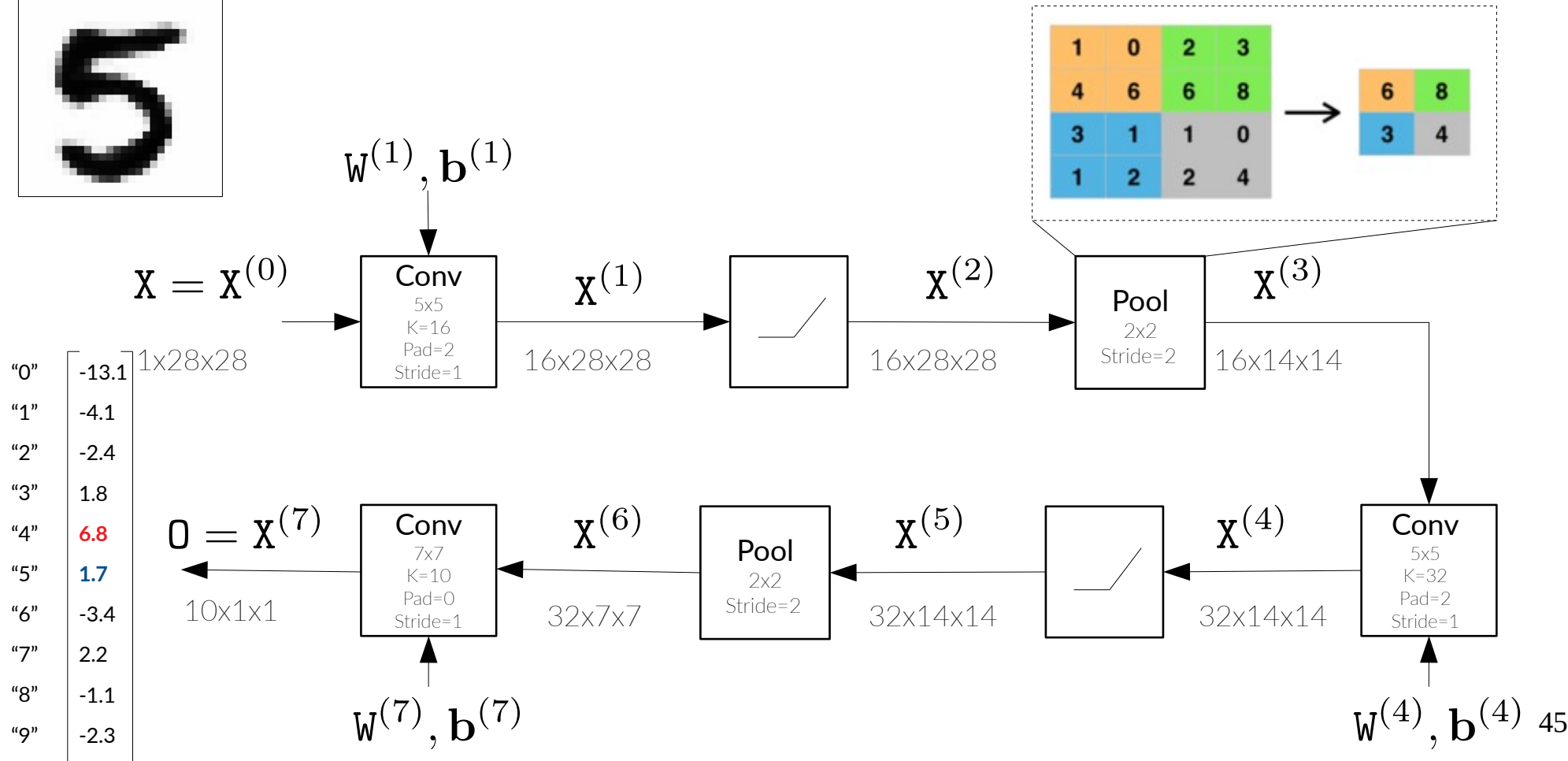
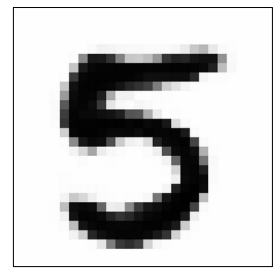


| | |
|-----|-------|
| "0" | -13.1 |
| "1" | -4.1 |
| "2" | -2.4 |
| "3" | 1.8 |
| "4" | 6.8 |
| "5" | 1.7 |
| "6" | -3.4 |
| "7" | 2.2 |
| "8" | -1.1 |
| "9" | -2.3 |



II)

Autre exemple d'architecture : CNN pour MNIST

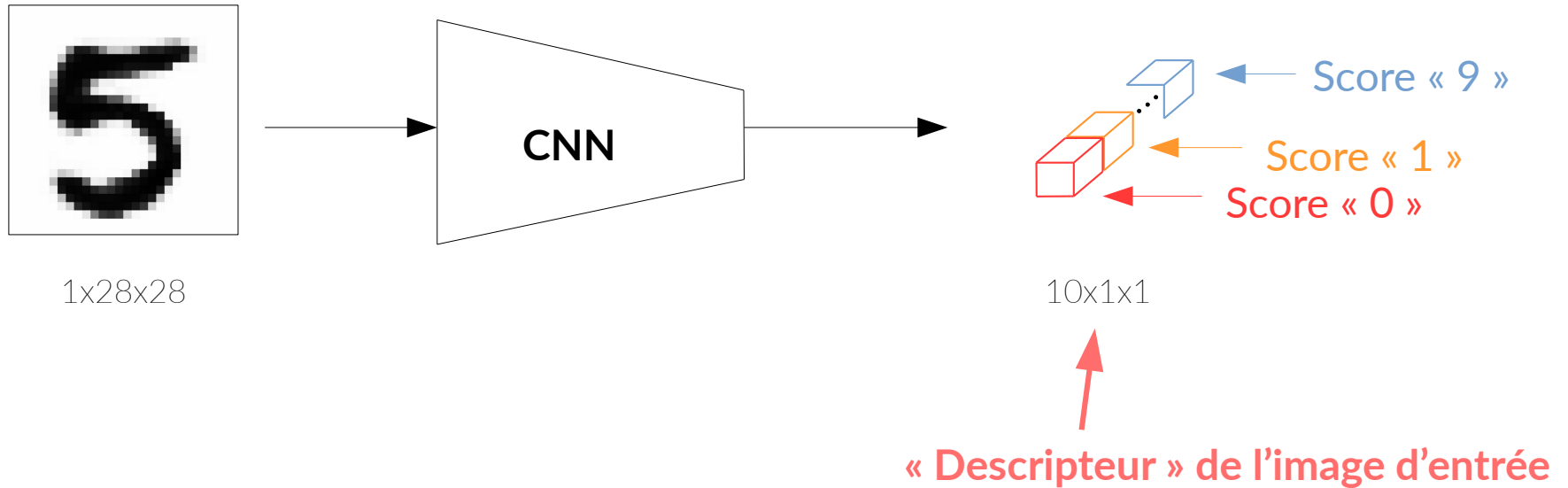


- "0" -13.1
- "1" -4.1
- "2" -2.4
- "3" 1.8
- "4" **6.8**
- "5" **1.7**
- "6" -3.4
- "7" 2.2
- "8" -1.1
- "9" -2.3

II)

Architectures de CNN : Deux cas extrêmes

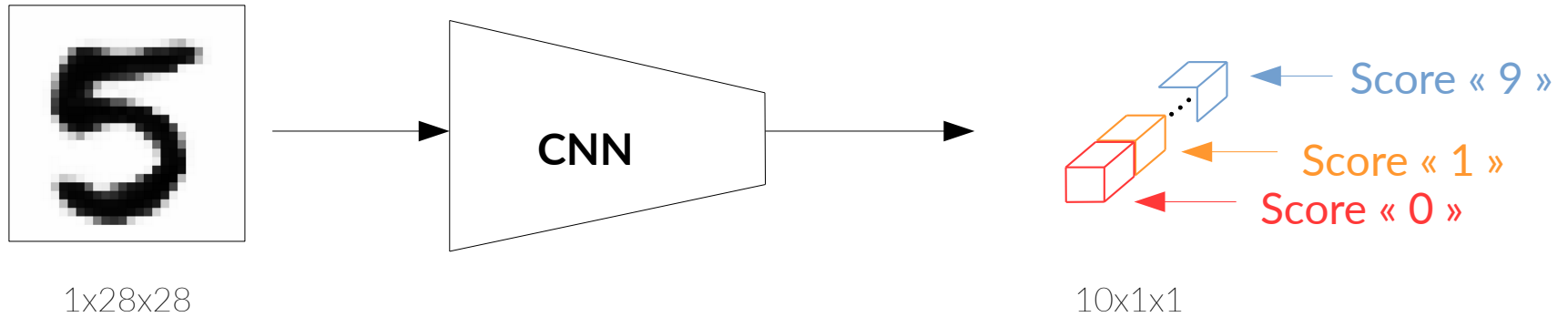
Cas 1 : Extraire une information **globale** présente dans l'image d'entrée



II)

Architectures de CNN : Deux cas extrêmes

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



Réduction **progressive** de la résolution



Utilisation de couches de conv ou pooling avec $\text{stride} = 2$

II)

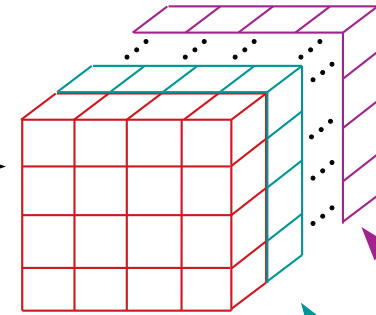
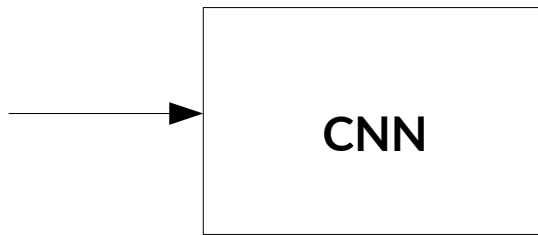
Architectures de CNN : Deux cas extrêmes

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



Cx480x640

Scores « Arbre »

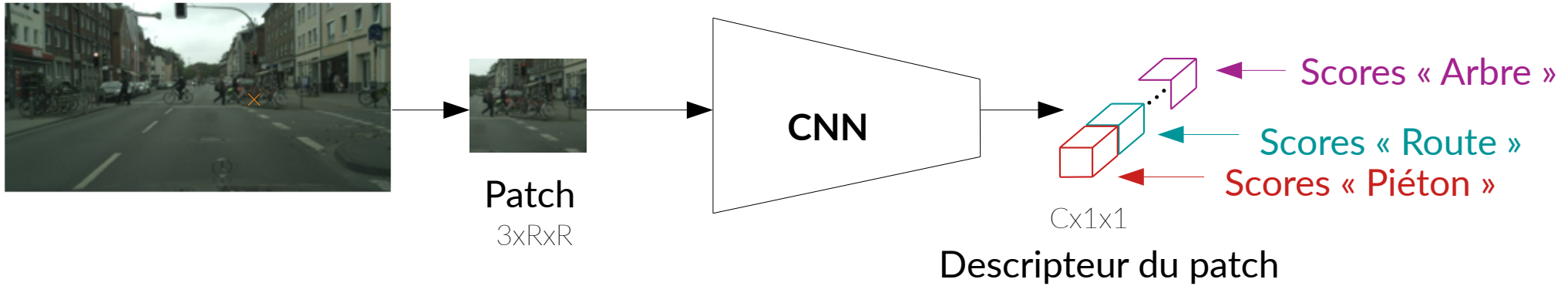
Scores « Route »

Scores « Piéton »

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

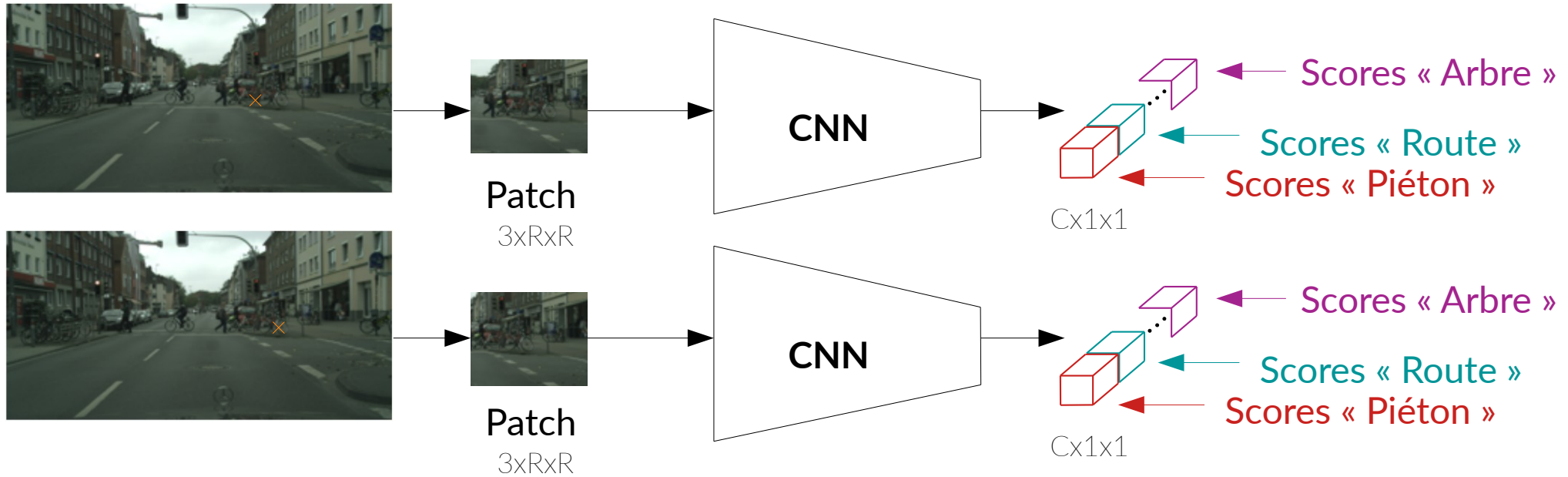
II)

Comment obtenir un descripteur pour chaque pixel ?



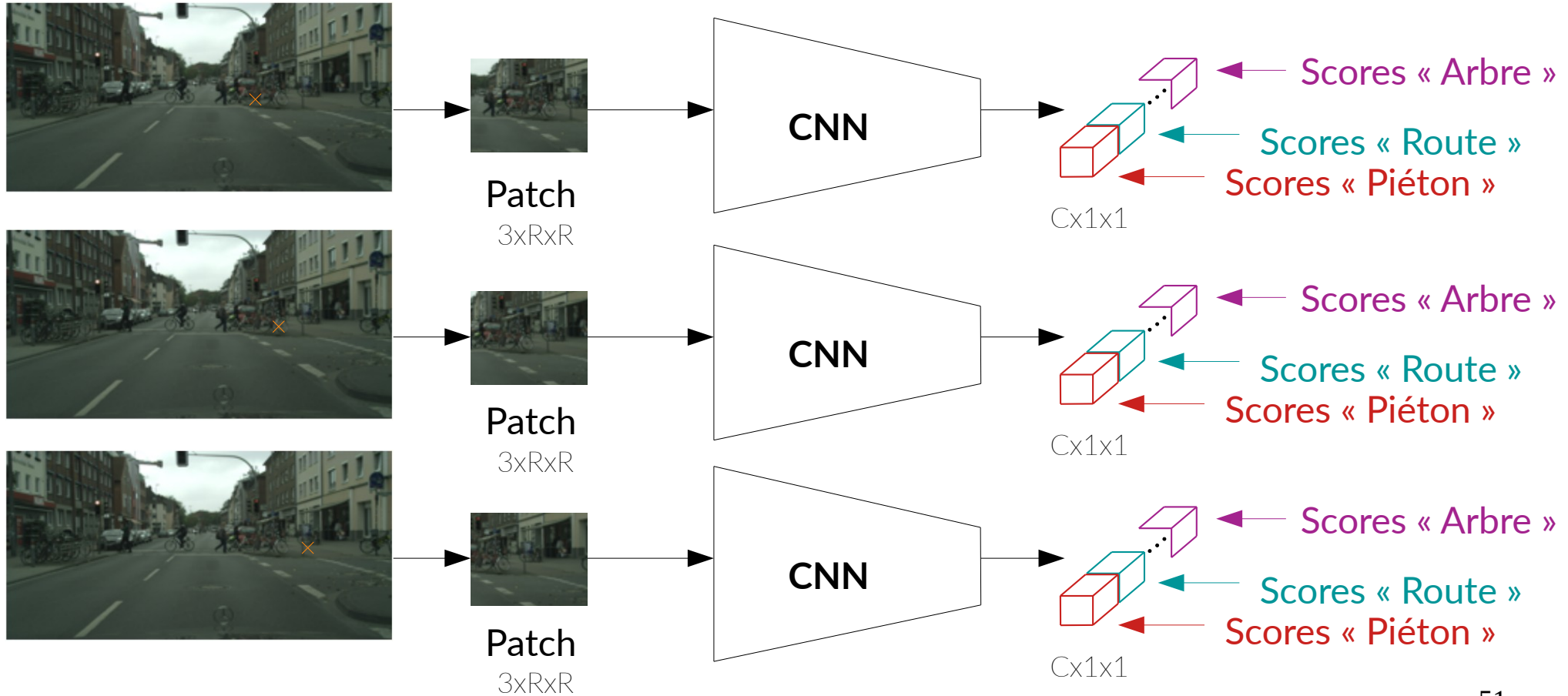
II)

Comment obtenir un descripteur pour chaque pixel ?



II)

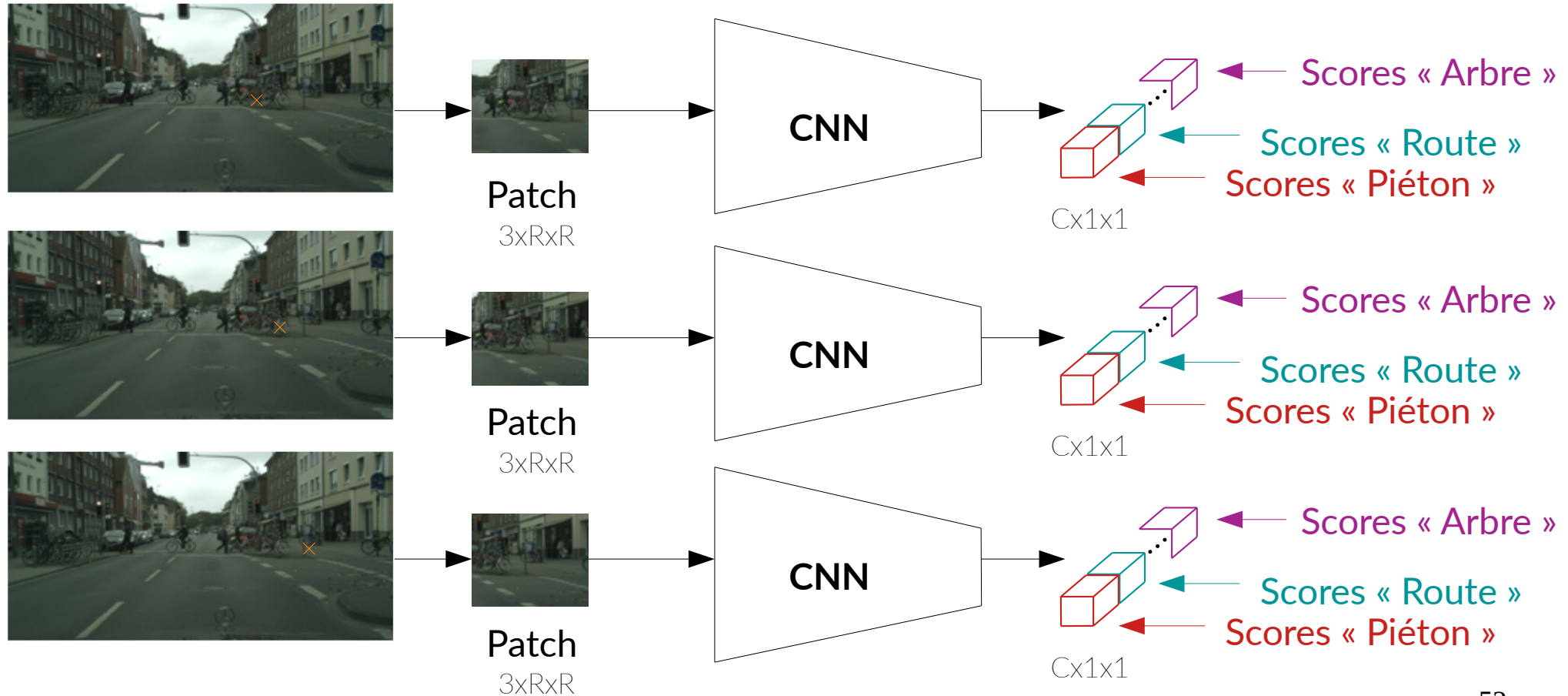
Comment obtenir un descripteur pour chaque pixel ?



Très inefficace en mémoire et en calculs !

II)

Comment obtenir un descripteur pour chaque pixel ?



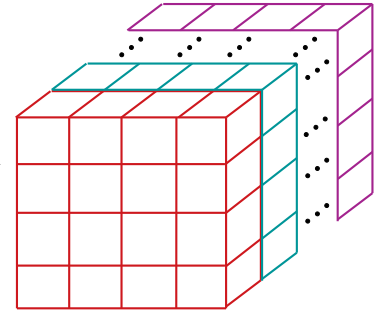
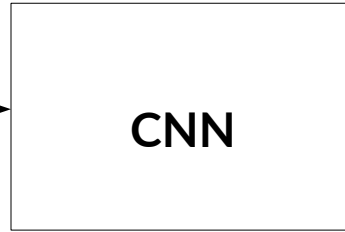
Très inefficace en mémoire et en calculs ! → Utilisation d'une architecture entièrement convolutive

II)

Architecture entièrement convolutive



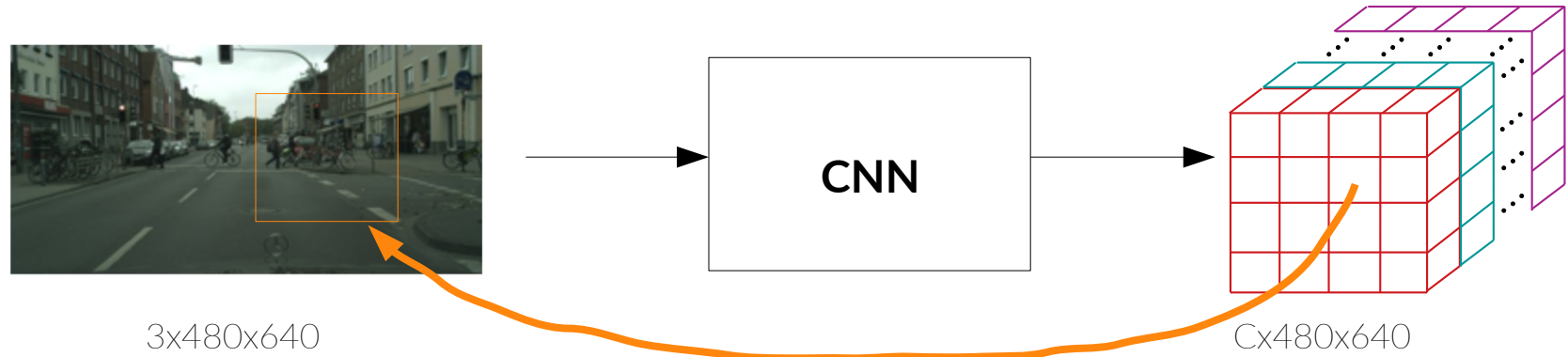
3x480x640



Cx480x640

II)

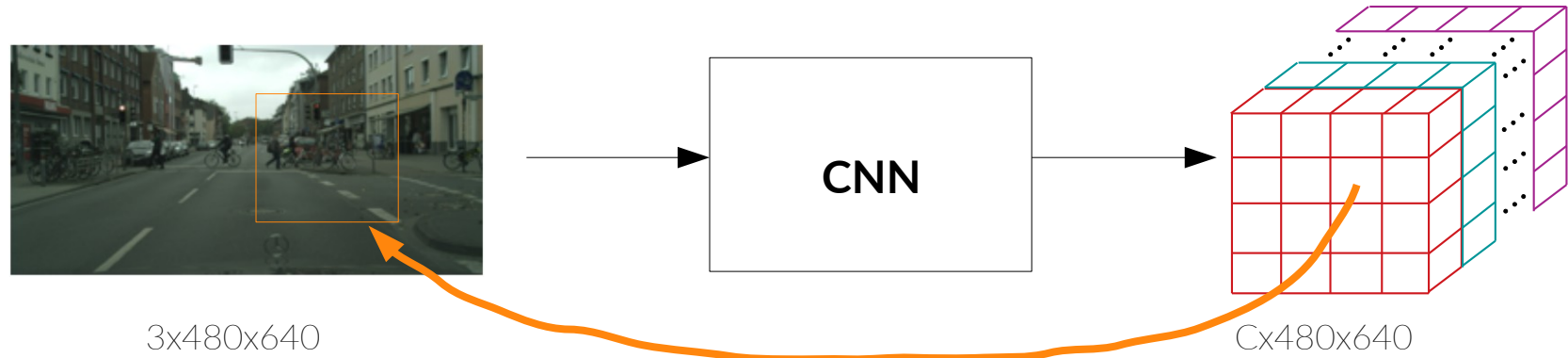
Architecture entièrement convolutive



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 »)

II)

Architecture entièrement convolutive

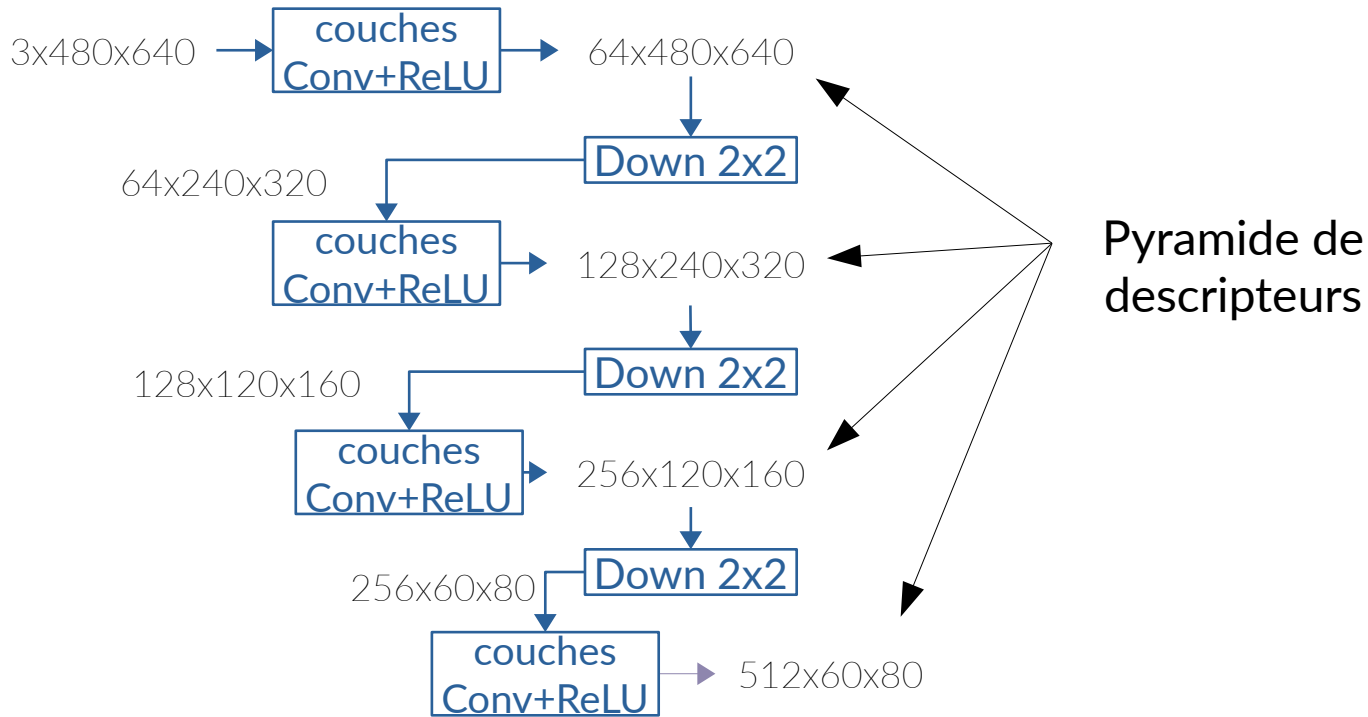


Descripteur $C \times 1 \times 1$ d'un patch de taille $R \times R$,
où $R \times R$ s'appelle le Champ Récepteur du CNN (« Receptive Field »)

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

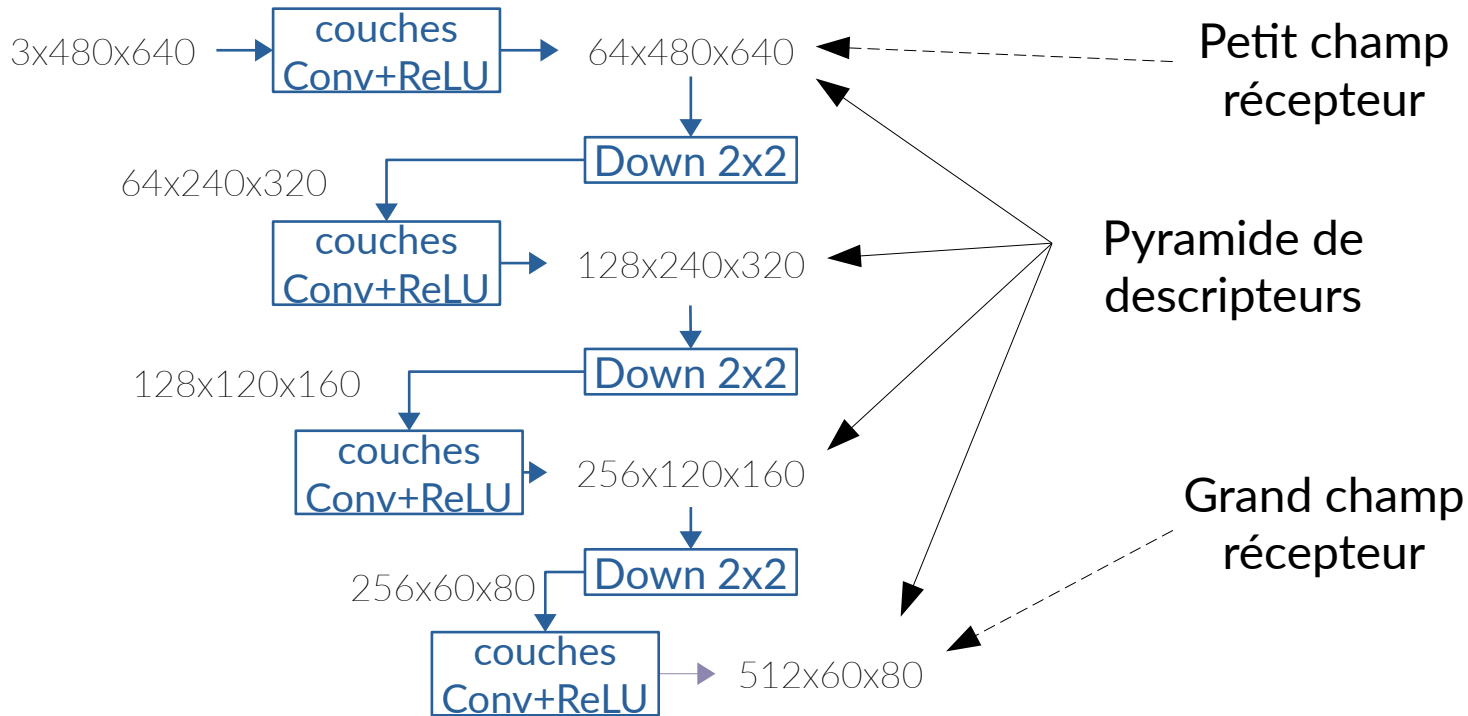
II)

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



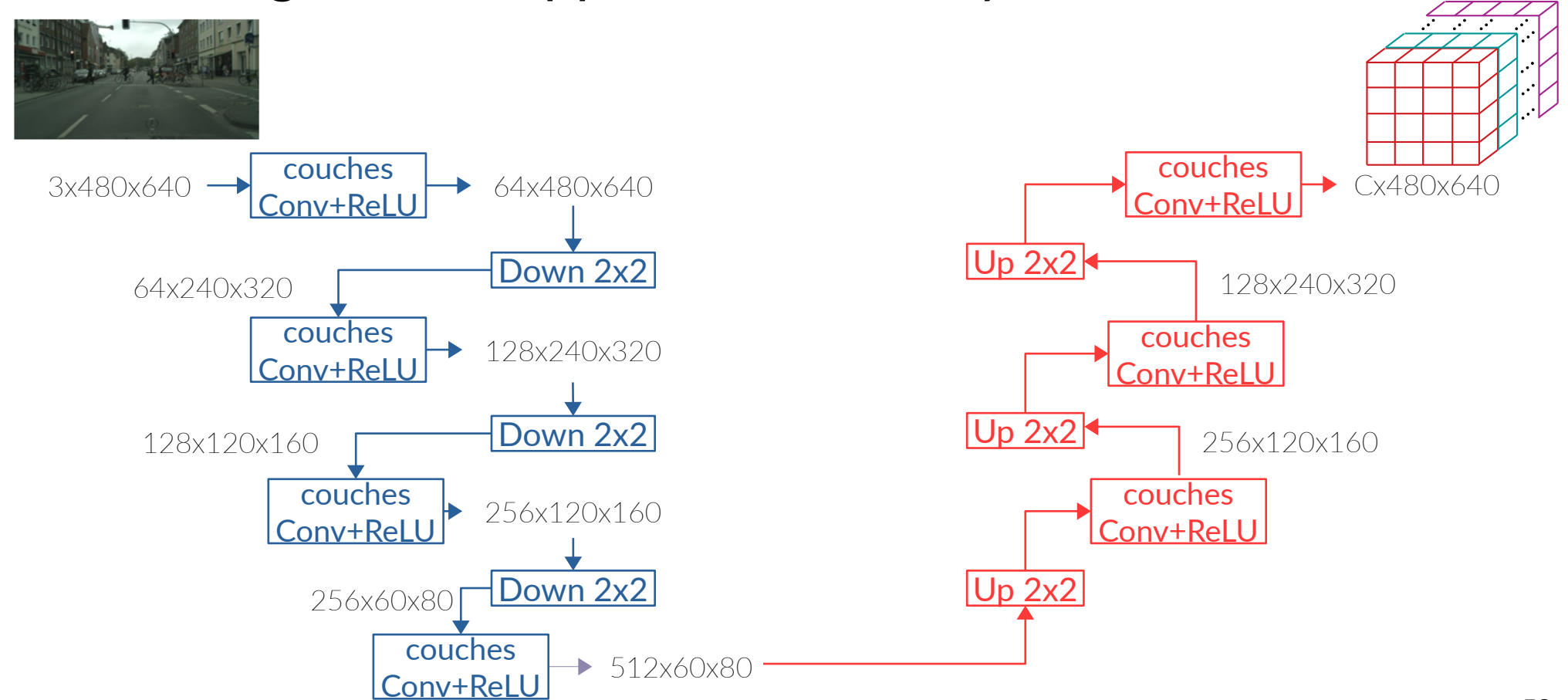
II)

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



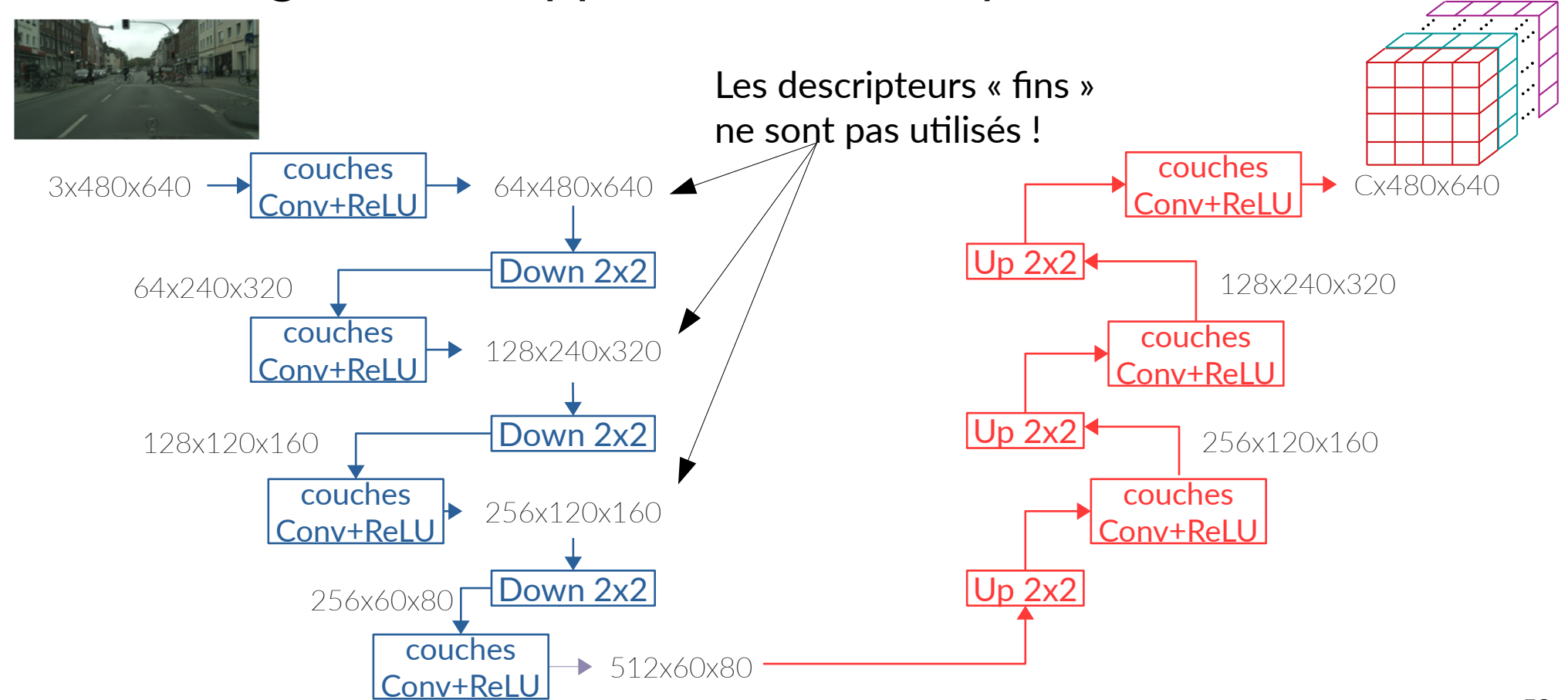
II)

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



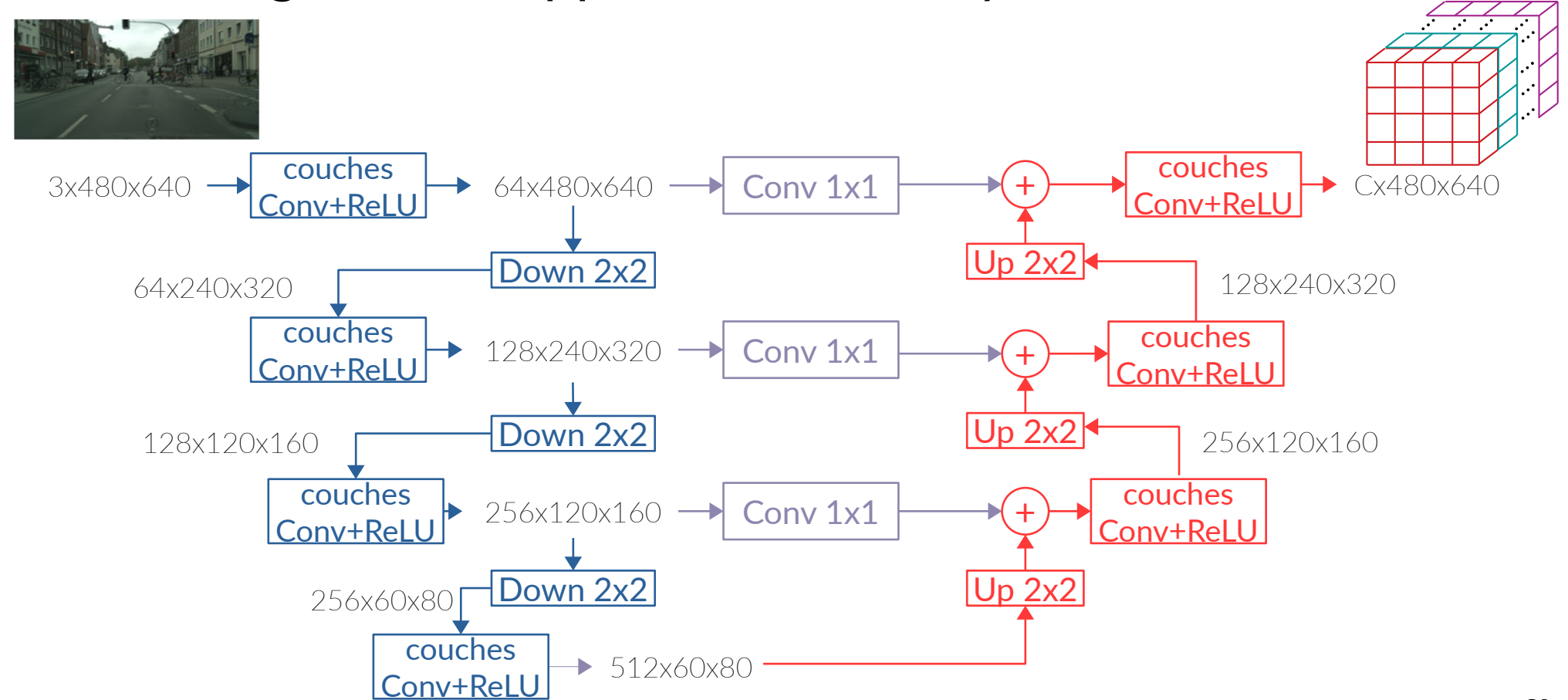
II)

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



II)

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



II)

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

