



IA et Nous *Réseaux convolutifs* *ou convolutionnels*

ConvNet CNN

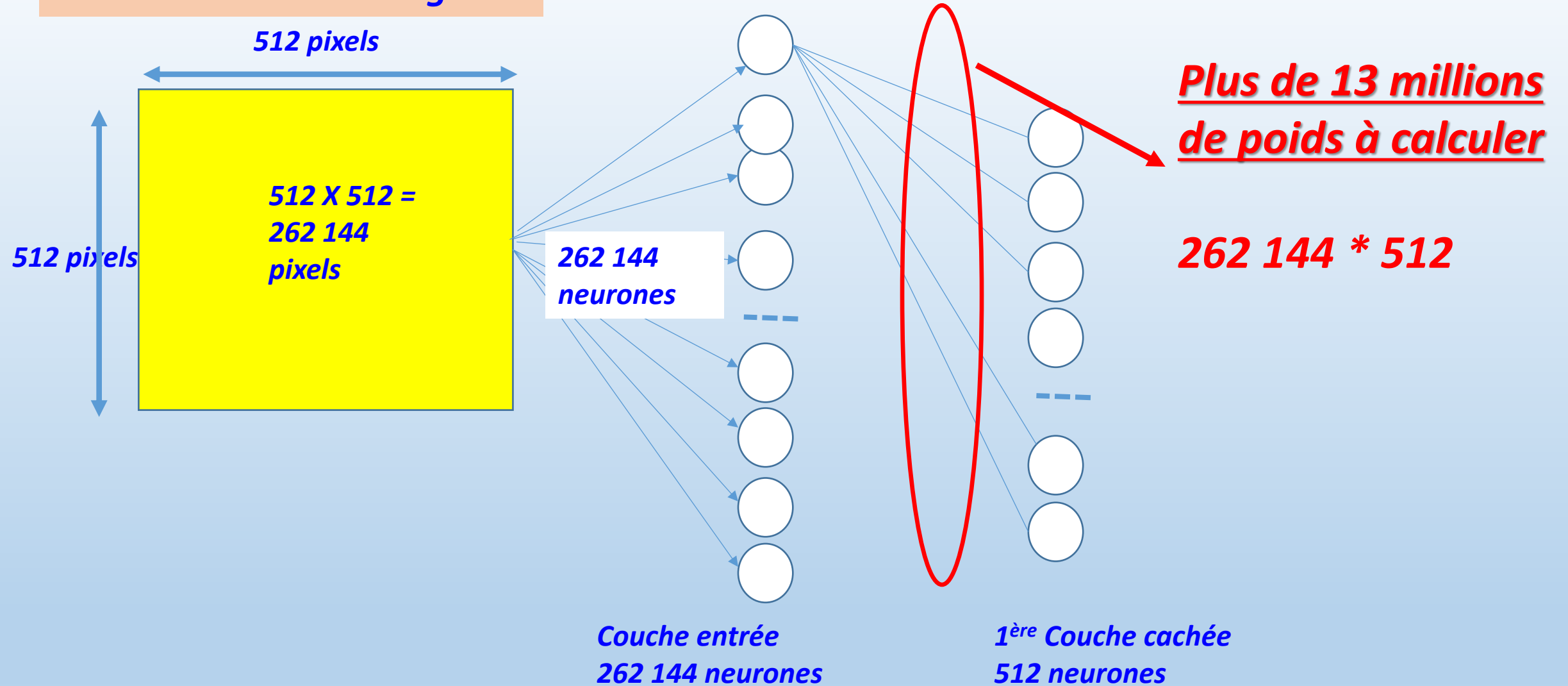
Les Réseaux convolutifs

Convolution Neuron Networks

- 1. Pourquoi ?*
- 2. Comment ça marche ?*
 - a. La Convolution*
 - b. Le Pooling*
 - c. Le Relu*
 - d. L'empilage de couches*
 - e. Les couches connectées- Classifieur*
 - f. L'entraînement*
- 3. Application à la reconnaissance des chiffres manuscrits*
- 4. Expliquer - Tester - Expérimenter*
- 5. Au-delà des images*

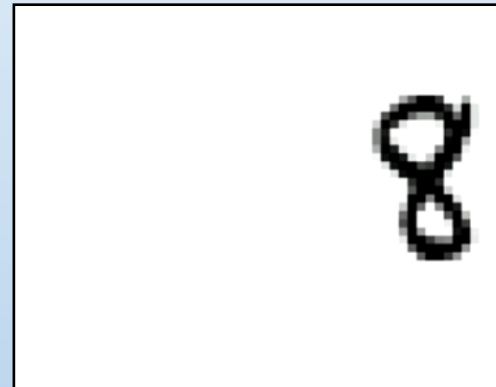
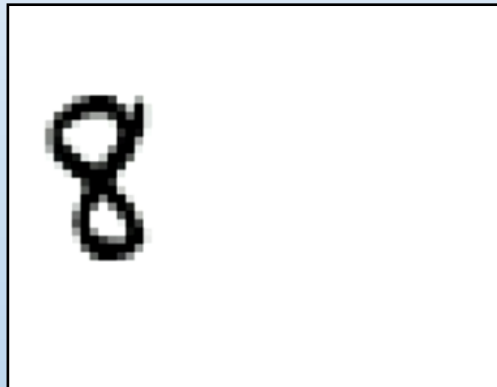
Pourquoi ?

Reconnaissance d'image



Pourquoi ?

- *Limites observées dans la lecture des chiffres manuscrits*
 - *Overfitting*
 - *Nombre de poids à calculer et à entraîner*
 - *Performances encore insuffisantes*
 - *Lenteur de l'apprentissage*
- *Invariance à la taille et à la translation*



Et si on apprenait différemment ?

Les CNN ou ConvNMet

Histoire

Le néocognitron , Kunihiko Fukushima en 1979

- basé sur des **modèles d'architecture neuronale** dans le cortex visuel des mammifères,
- a servi de base aux **réseaux de neurones convolutifs** (CNN),

Les CNN : Convolutional Neuron Networks *

Fers de lance de l'Apprentissage profond

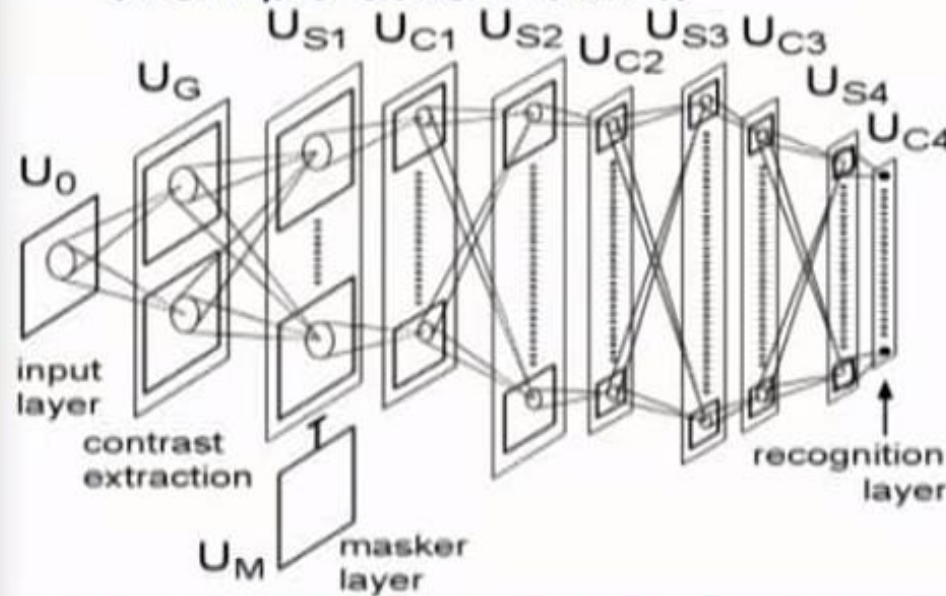
*** Popularisés par Yann Lecun**

Neocognitron

Hubel & Wiesel's Model of the Architecture of the Visual Cortex

■ [Hubel & Wiesel 1962]:

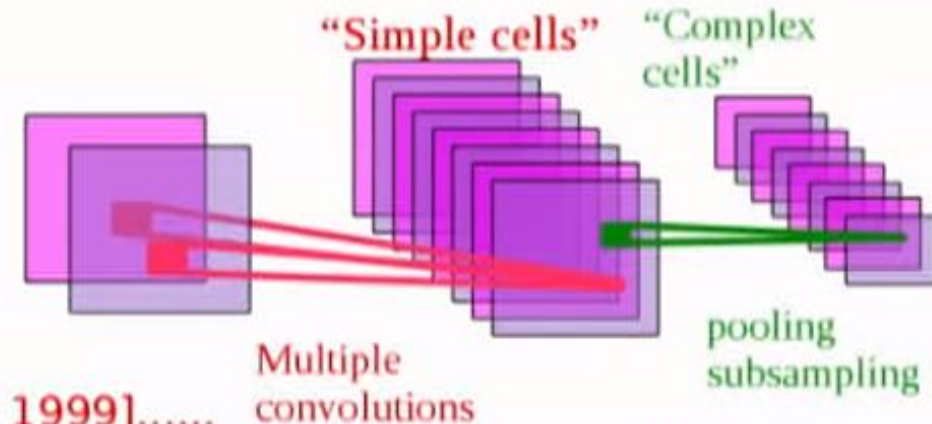
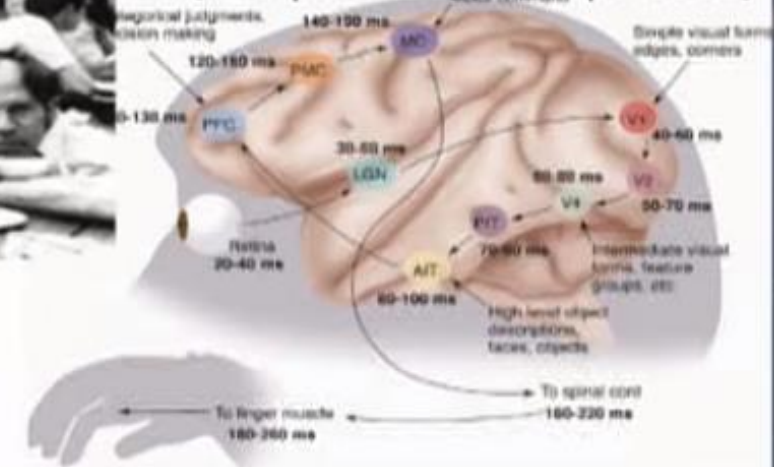
- ▶ **simple cells** detect local features
- ▶ **complex cells** “pool” the outputs of simple cells within a



[Fukushima 1982][LeCun 1989, 1998],[Riesenhuber 1999].....



[Thorpe & Fabre-Thorpe 2001]



Le monde est compositionnel

Représentation **hiérarchique** avec un niveau croissant d'abstraction :

- Image : pixel - bord - contour arrangé - motif (cercle, coin...) – partie d'objet – objet
- Texte : caractère – mot - groupe de mots – phrase – histoire
- Parole : échantillon élémentaire – son phonème – mot - ...

Yann Lecun : « Comme sa grande sœur biologique, l'intelligence artificielle appréhende la réalité extérieure de façon **hiérarchique**, en voyant "d'abord des contours arrangés, puis des motifs comme des cercles ou des coins, puis des parties d'objets ».

Comment ça marche ?

Comment ça marche ?

Principe

En "glissant" une petite couche sur une entrée plus grande, un CNN peut effectuer un traitement plus approfondi avec moins de calculs.

Par exemple, une image **100 × 100** soit **10 000 pixels**,

→ **10 000 poids** à traiter avec une couche **entièrement connectée** ;

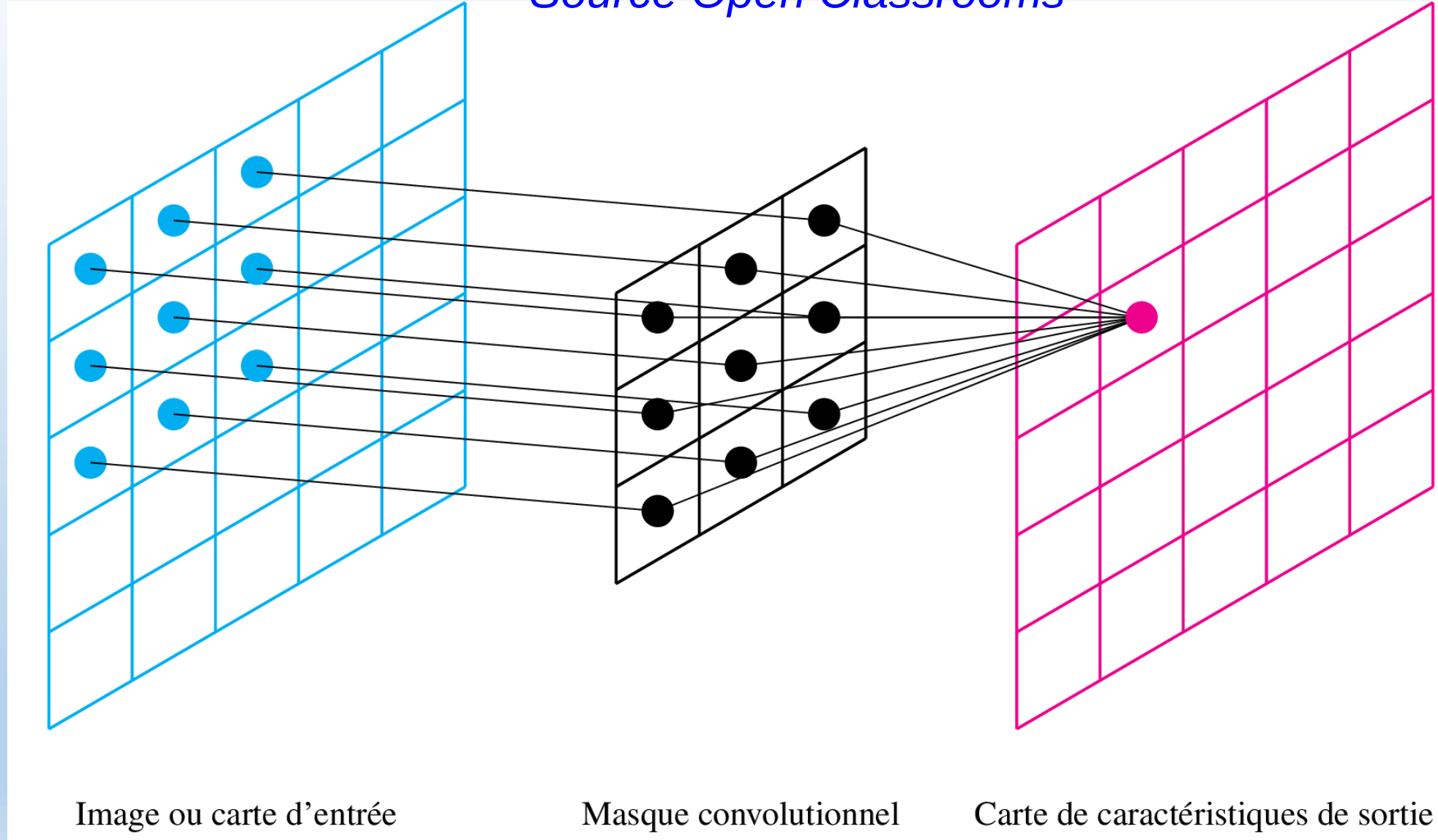
Une **couche convolutive** constituée d'une "**fenêtre**" **5 × 5 glissant** sur l'image peut effectuer une détection des contours

→ **25 paramètres apprenables.**

Les couches convolutives sont combinées par des "**couches de regroupement**" et traitées par des couches "**entièrement connectées**" qui sont généralement **des perceptrons multicouches**

Les CNN ou ConvNet

Source Open Classrooms



Les CNN ou ConvNet

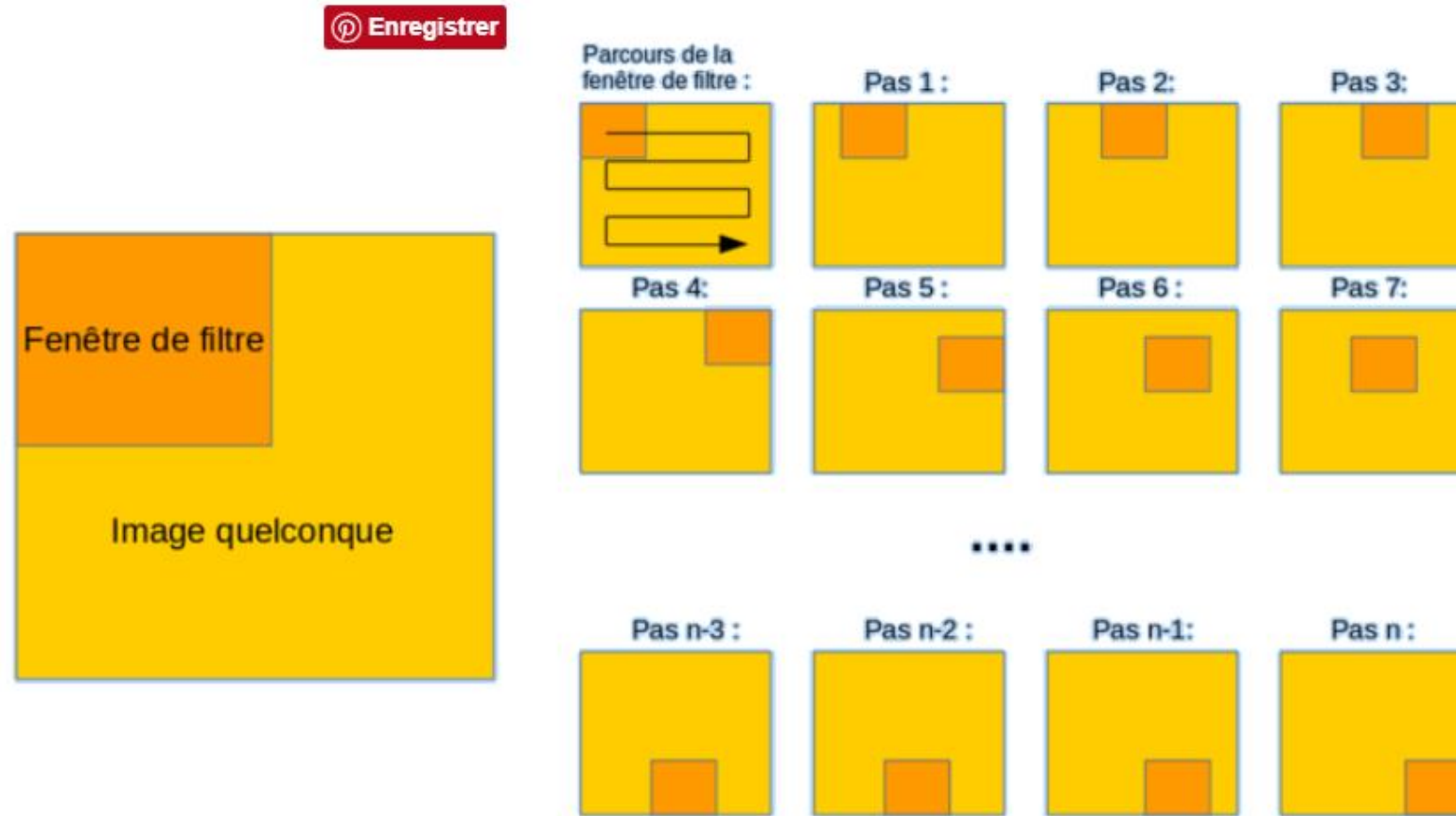
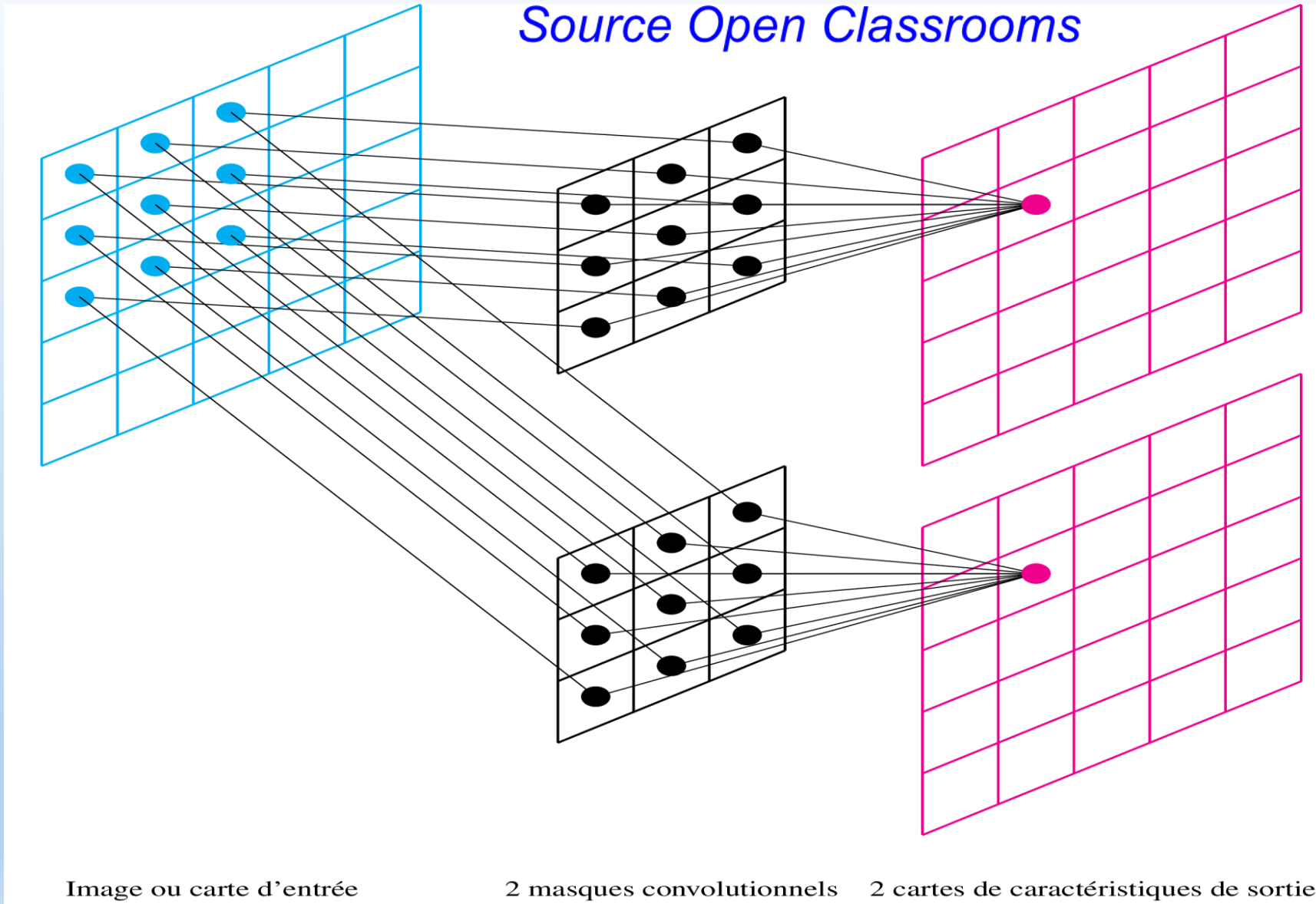


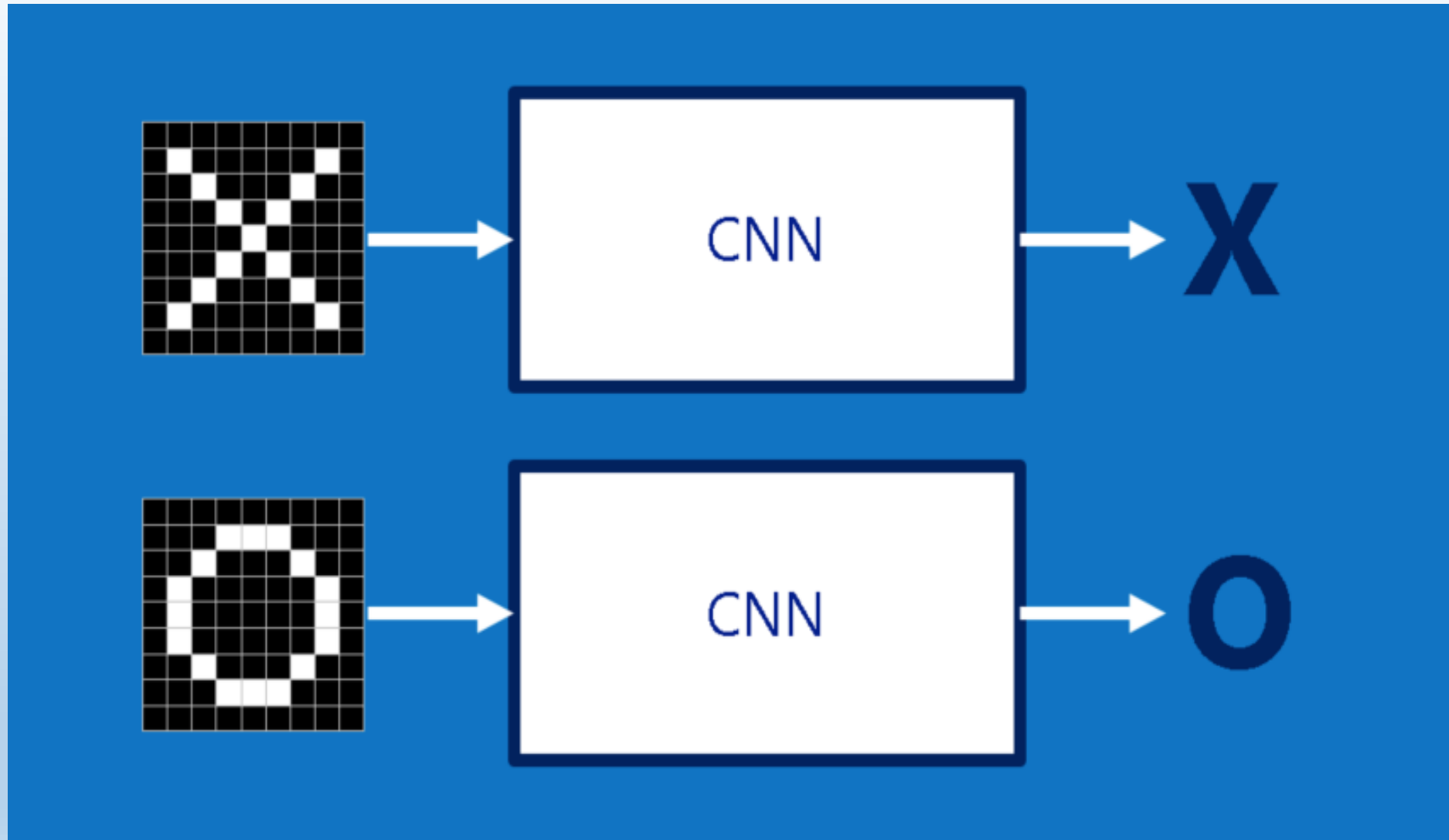
Schéma du parcours de la fenêtre de filtre sur l'image.

Les CNN ou ConvNet



Comment ça marche ?

Tiré d'un article de [Brandon Rohrer](#) (Senior Data Scientist à Facebook), "[How Convolutional Neural Networks Work](#)"



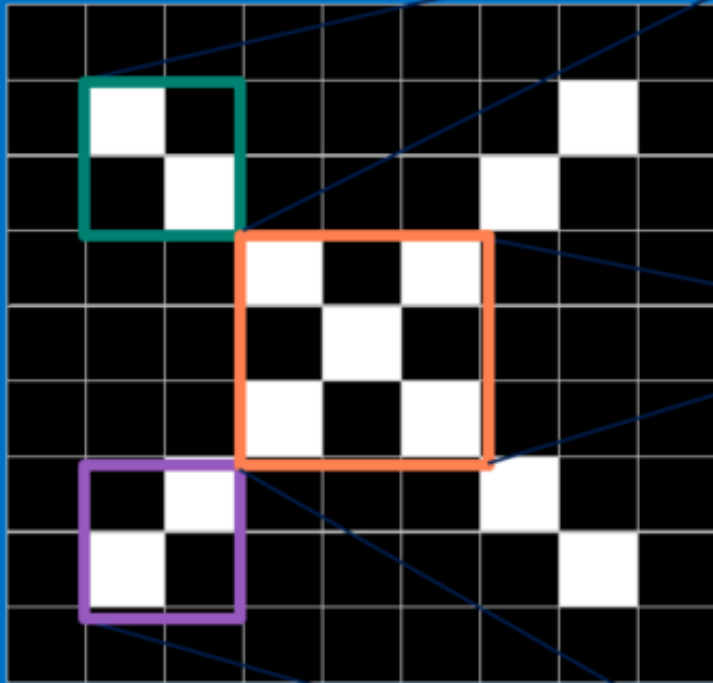
Comparaison d'images

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1



-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	1	-1	-1	-1
-1	-1	1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Ou recherche de caractéristiques

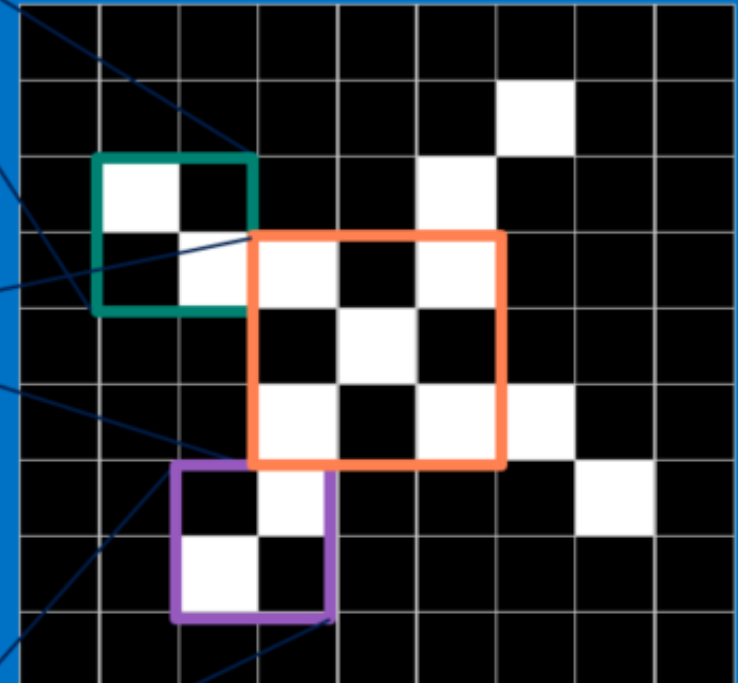


Recherche de
Caractéristiques
(features)

=

=

=

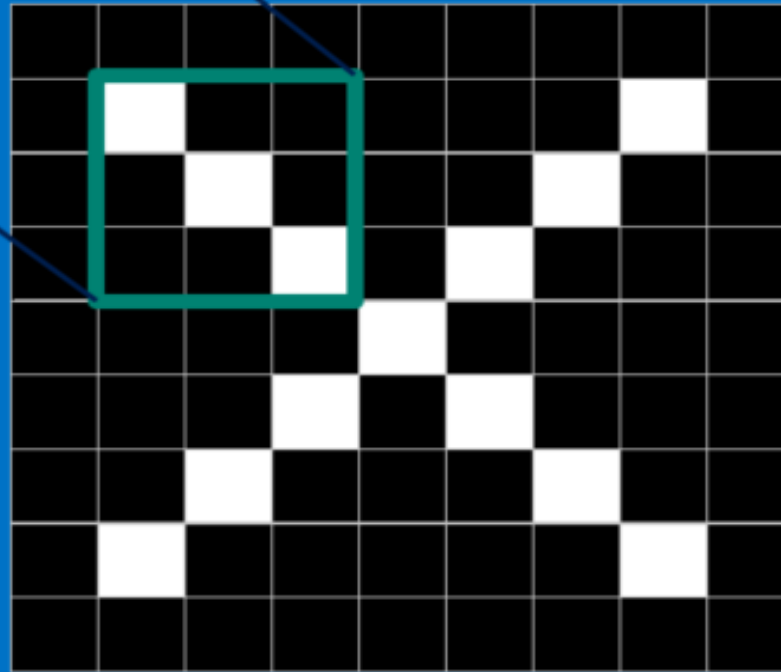


Les 3 caractéristiques

1	-1	-1
-1	1	-1
-1	-1	1

1	-1	1
-1	1	-1
1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1



La convolution

1	-1	-1
-1	1	-1
-1	-1	1

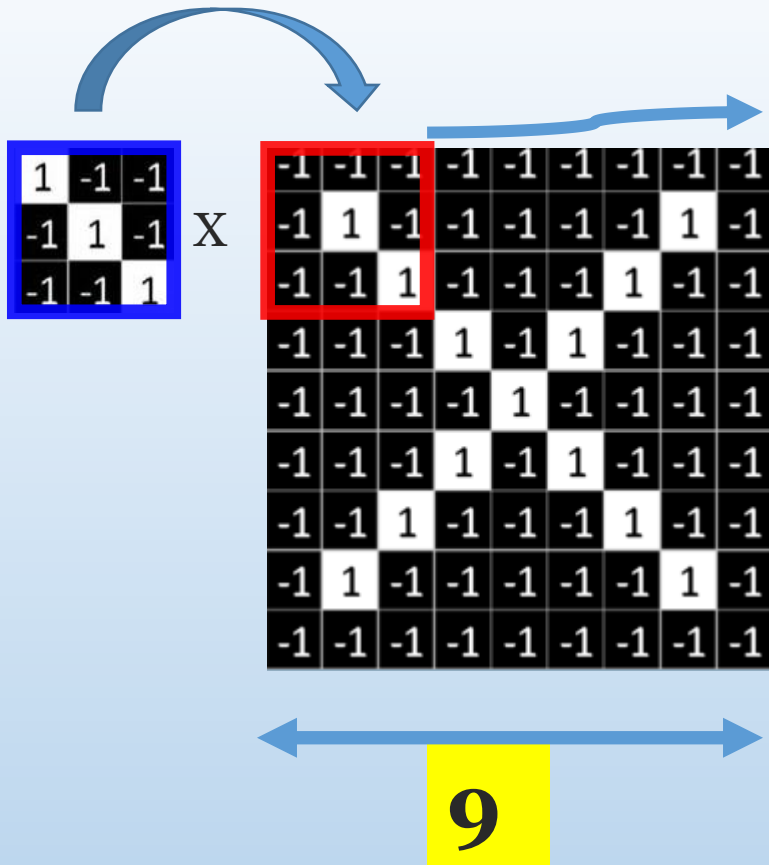
$$1 \times 1 = 1$$

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	1
1	1	1
1	1	1

Si concordance : + 1
Si décalage : - 1

La convolution



-1	1	1
1	1	1
1	1	1

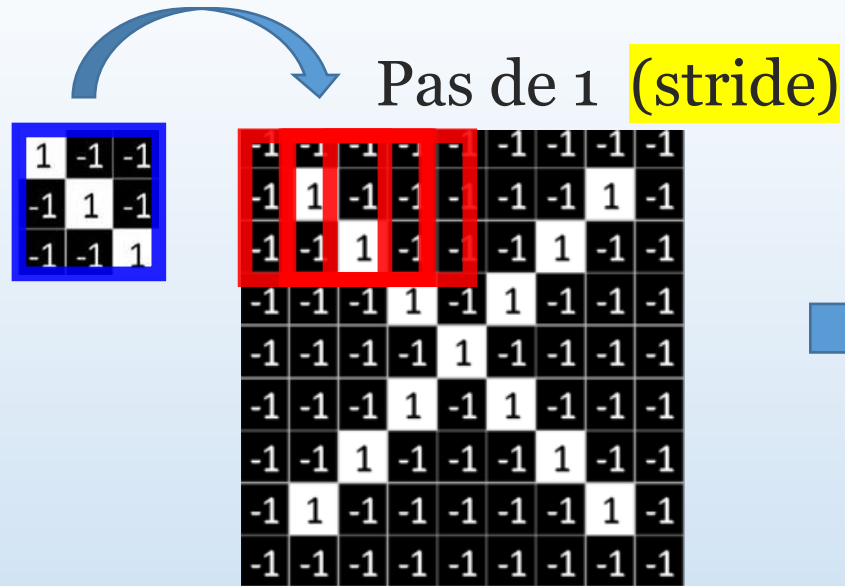
$$\begin{aligned}
 1 * (-1) + (-1) * (-1) + (-1) * (-1) &= 1 \\
 (-1) * (-1) + 1 * 1 + (-1) * (-1) &= 3 \\
 (-1) * (-1) + (-1) * (-1) + (-1) * 1 &= 3
 \end{aligned}$$

Soit $7/9 = 0,77$

Diagram showing the 7x7 output grid with values ranging from -0.11 to 1.00. A yellow box with the number 7 is at the bottom, indicating the output size.

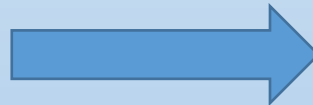
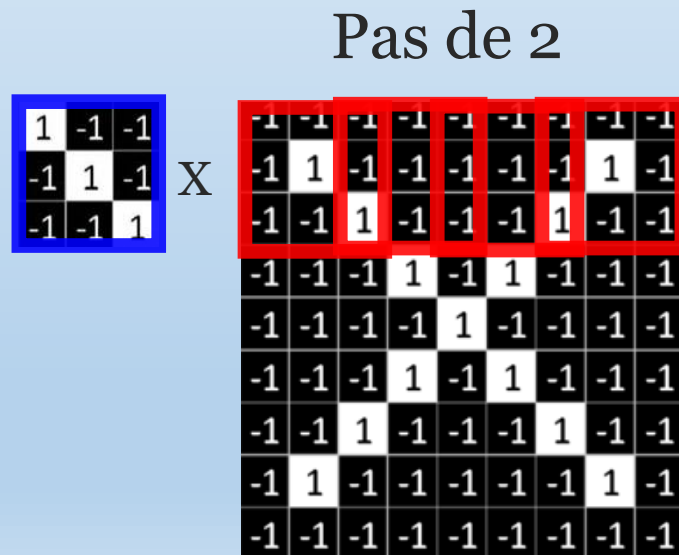
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

La convolution



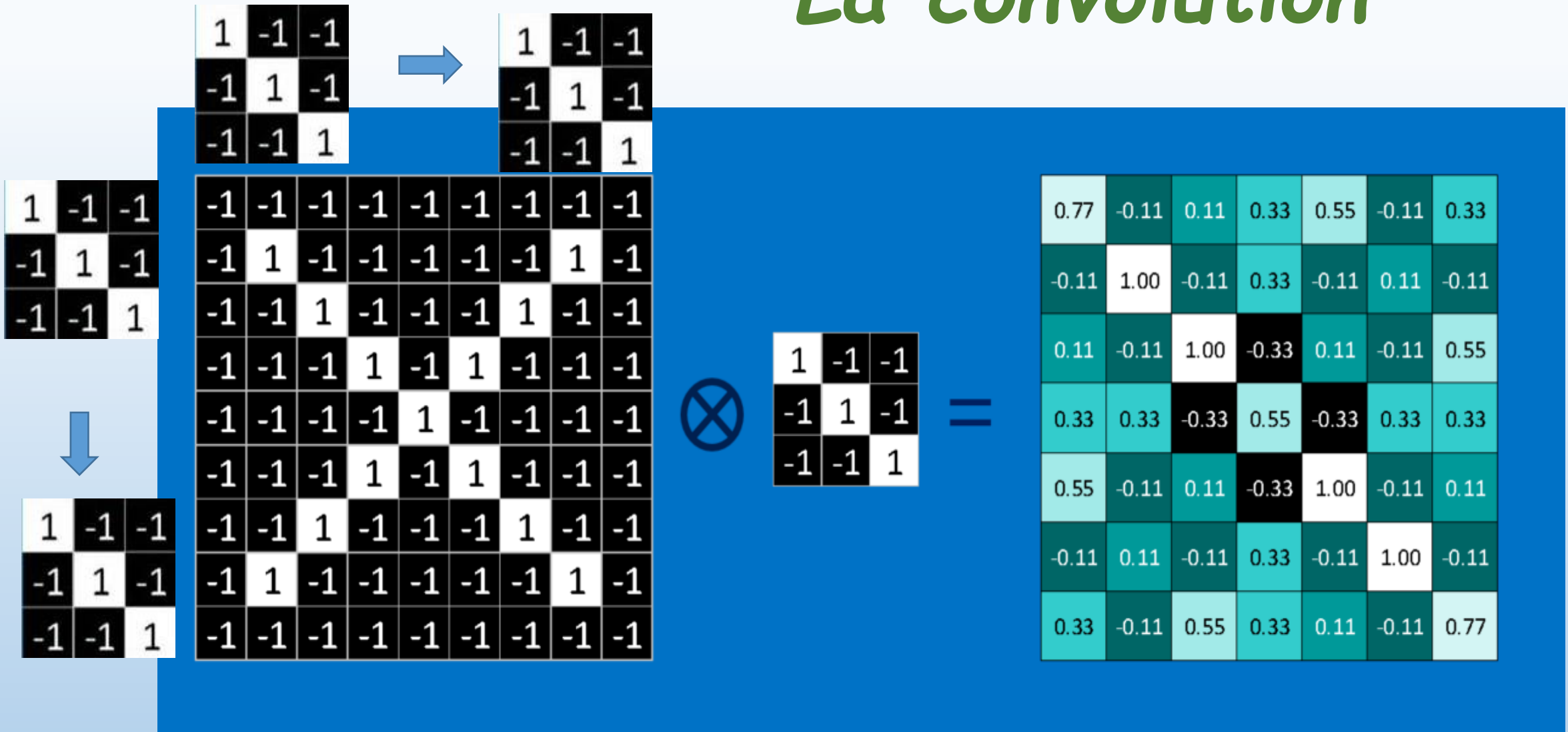
Plan de 8 X 8

*Avec traitement des bords
Padding*



Plan de 4 X 4

La convolution



Couche de convolution

On répète ce processus pour les 3 caractéristiques

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1



0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

0.33	-0.33	0.11	-0.11	0.11	-0.33	0.33
-0.33	0.33	-0.33	0.33	-0.33	0.33	-0.33
0.11	-0.33	0.33	-0.77	0.33	-0.33	0.11
-0.11	0.33	-0.77	1.00	-0.77	0.33	-0.11
0.11	-0.33	0.33	-0.77	0.33	-0.33	0.11
-0.33	0.33	-0.33	0.33	-0.33	0.33	-0.33
0.33	-0.33	0.11	-0.11	0.11	-0.33	0.33

0.33	-0.11	0.33	0.33	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.11	-0.33	1.00	-0.11	0.11
0.33	0.33	-0.33	0.33	-0.33	0.33	0.33
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.33	0.33	-0.11	0.33

Images filtrées

Le Pooling

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

maximum

1.00			

**Réduction de la taille des Images
filtrées tout en conservant les
informations importantes**

Le Pooling

0.77	-0.11	0.11	0.25	0.25	-0.11	0.25
-0.11	1.00	-0.11	0.25	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.25	0.11	-0.11	0.25
0.25	0.25	-0.25	0.25	-0.25	0.25	0.25
0.25	-0.11	0.11	-0.25	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.25	-0.11	1.00	-0.11
0.25	-0.11	0.25	0.25	0.11	-0.11	0.77

0.33	-0.25	0.11	-0.11	0.11	-0.25	0.33
-0.25	0.25	-0.25	0.33	-0.25	0.25	-0.25
0.11	-0.25	0.25	-0.77	0.25	-0.25	0.11
-0.11	0.33	-0.77	1.00	-0.77	0.33	-0.11
0.11	-0.25	0.25	-0.77	0.25	-0.25	0.11
-0.25	0.25	-0.25	0.33	-0.25	0.25	-0.25
0.33	-0.25	0.11	-0.11	0.11	-0.25	0.33

0.33	-0.11	0.25	0.33	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.25	-0.11	0.11	-0.25	1.00	-0.11	0.11
0.33	0.33	-0.33	0.25	-0.33	0.33	0.33
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.25
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.33	0.25	-0.11	0.33

1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

0.55	0.33	0.55	0.33
0.33	1.00	0.55	0.11
0.55	0.55	0.55	0.11
0.33	0.11	0.11	0.33

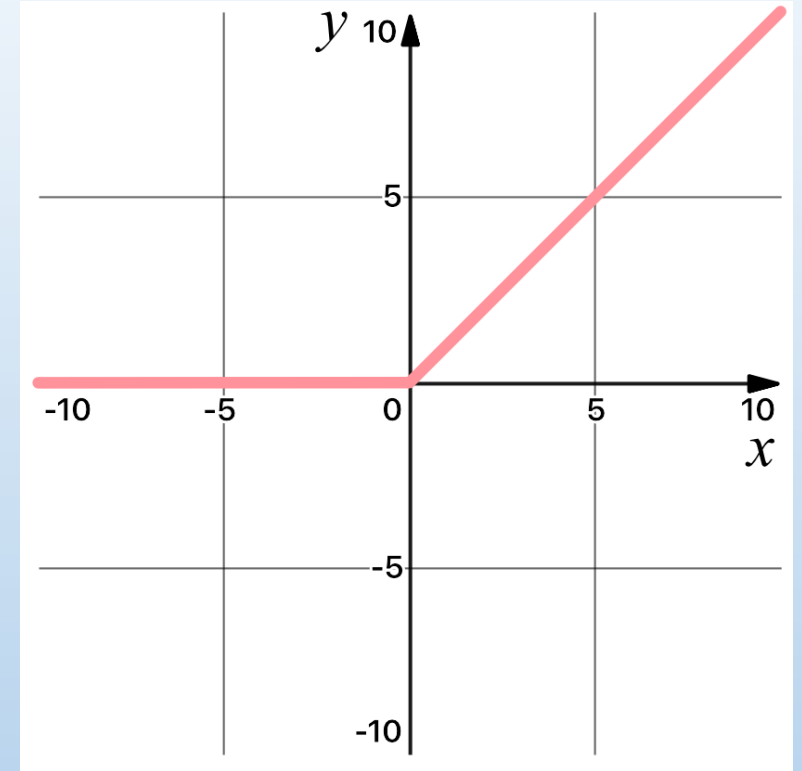
0.33	0.55	1.00	0.77
0.55	0.55	1.00	0.33
1.00	1.00	0.11	0.55
0.77	0.33	0.55	0.33

Unité Rectifiée Linéaire : Le ReLu

La fonction d'activation ReLU est spécifiquement utilisée comme fonction d'activation non linéaire

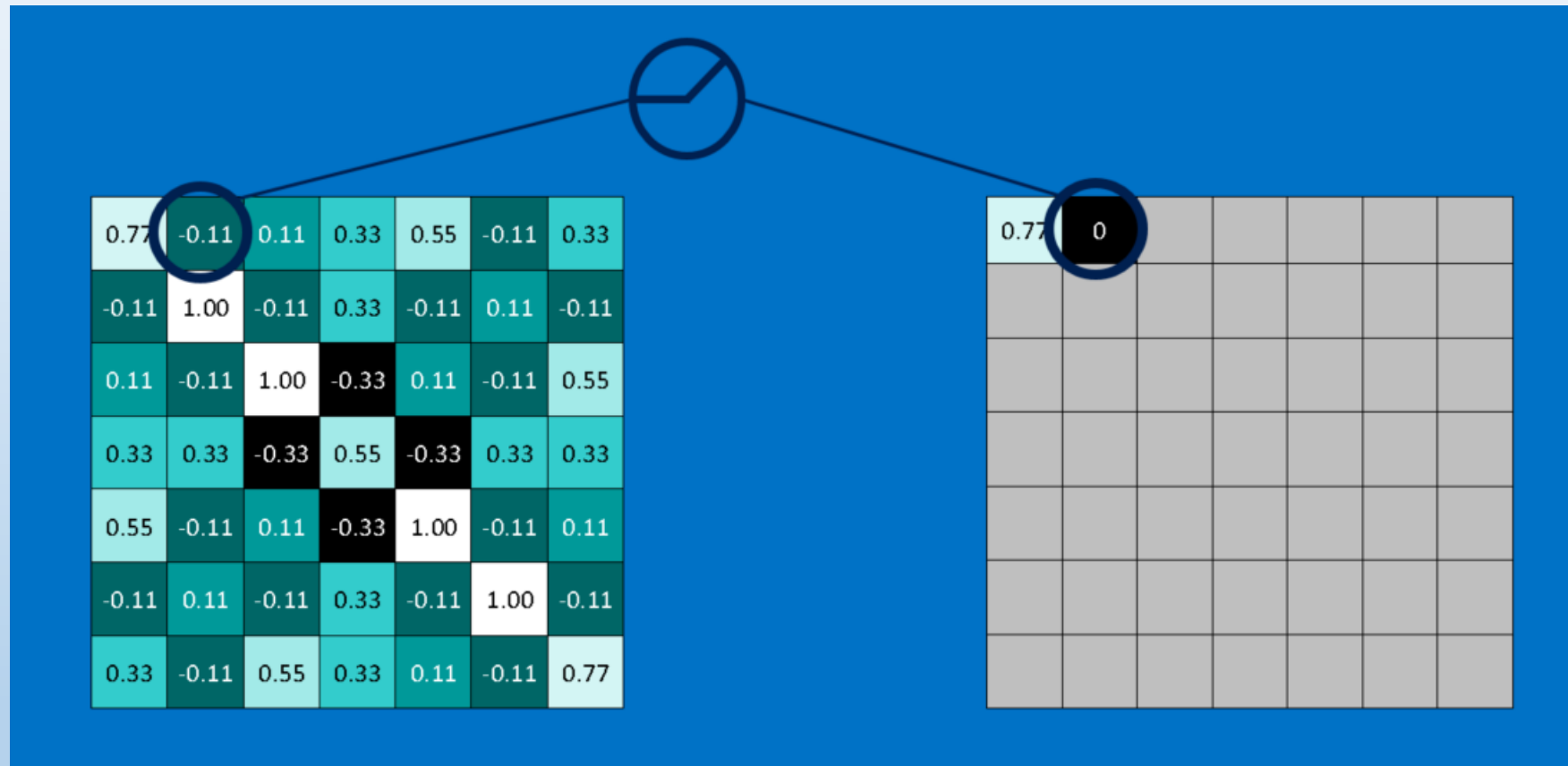
*Préférée aux autres fonctions non linéaires telles que Sigmoides, car il a été observé de manière empirique que les **CNN utilisant ReLU sont plus rapides** à former que leurs homologues.*

NB : ReLu correspondrait mieux au schéma d'activation du neurone biologique.



Unité Rectifiée Linéaire : Le ReLu

L'opération consiste à remplacer les valeurs négatives pour un pixel par zéro
Cela empêche les valeurs apprises de rester coincées à 0 ou de tendre vers l'infini
Garder le CNN en « bonne santé » mathématiquement



Après ReLu

0.77	-0.11	0.11	0.25	0.25	-0.11	0.25
-0.11	1.00	-0.11	0.25	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.25	0.11	-0.11	0.25
0.25	0.25	-0.25	0.25	-0.25	0.25	0.25
0.25	-0.11	0.11	-0.25	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.25	-0.11	1.00	-0.11
0.25	-0.11	0.25	0.25	0.11	-0.11	0.77

0.25	-0.25	0.11	-0.11	0.11	-0.25	0.25
-0.25	0.25	-0.25	0.25	-0.25	0.25	-0.25
0.11	-0.25	0.25	-0.77	0.25	-0.25	0.11
-0.11	0.25	-0.77	1.00	-0.77	0.25	-0.11
0.11	-0.25	0.25	-0.77	0.25	-0.25	0.11
-0.25	0.25	-0.25	0.25	-0.25	0.25	-0.25
0.25	-0.25	0.11	-0.11	0.11	-0.25	0.25

0.25	-0.11	0.25	0.25	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.25	-0.11	1.00	-0.11
0.25	-0.11	0.11	-0.25	1.00	-0.11	0.11
0.25	0.25	-0.25	0.25	-0.25	0.25	0.25
0.11	-0.11	1.00	-0.25	0.11	-0.11	0.25
-0.11	1.00	-0.11	0.25	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.25	0.25	-0.11	0.25

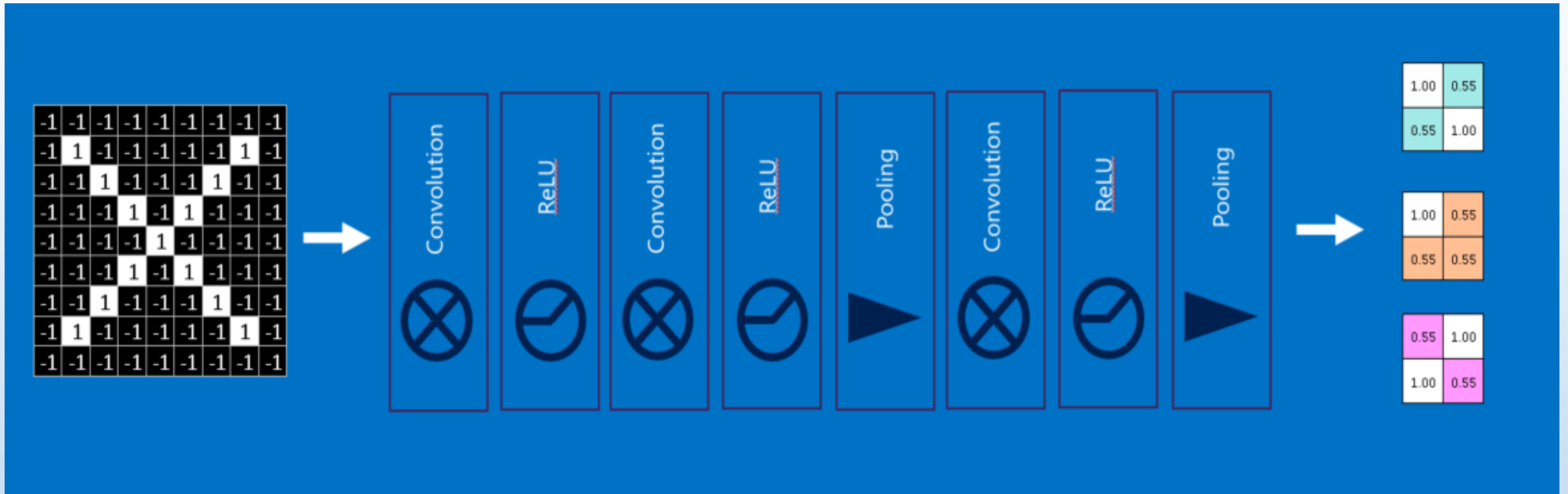


0.77	0	0.11	0.25	0.25	0	0.25
0	1.00	0	0.25	0	0.11	0
0.11	0	1.00	0	0.11	0	0.25
0.25	0.25	0	0.25	0	0.25	0.25
0.25	0	0.11	0	1.00	0	0.11
0	0.11	0	0.25	0	1.00	0
0.25	0	0.25	0.25	0.11	0	0.77

0.25	0	0.11	0	0.11	0	0.25
0	0.25	0	0.25	0	0.25	0
0.11	0	0.25	0	0.25	0	0.11
0	0.25	0	1.00	0	0.25	0
0.11	0	0.25	0	0.25	0	0.11
0	0.25	0	0.25	0	0.25	0
0.25	0	0.11	0	0.11	0	0.25

0.25	0	0.25	0.25	0.11	0	0.77
0	0.11	0	0.25	0	1.00	0
0.25	0	0.11	0	1.00	0	0.11
0.25	0.25	0	0.25	0	0.25	0.25
0.11	0	1.00	0	0.11	0	0.25
0	1.00	0	0.25	0	0.11	0
0.77	0	0.11	0.25	0.25	0	0.25

Empilage de couches



On empile les couches

A chaque fois, les caractéristiques deviennent plus grandes et plus complexes, et les images deviennent plus compactes.

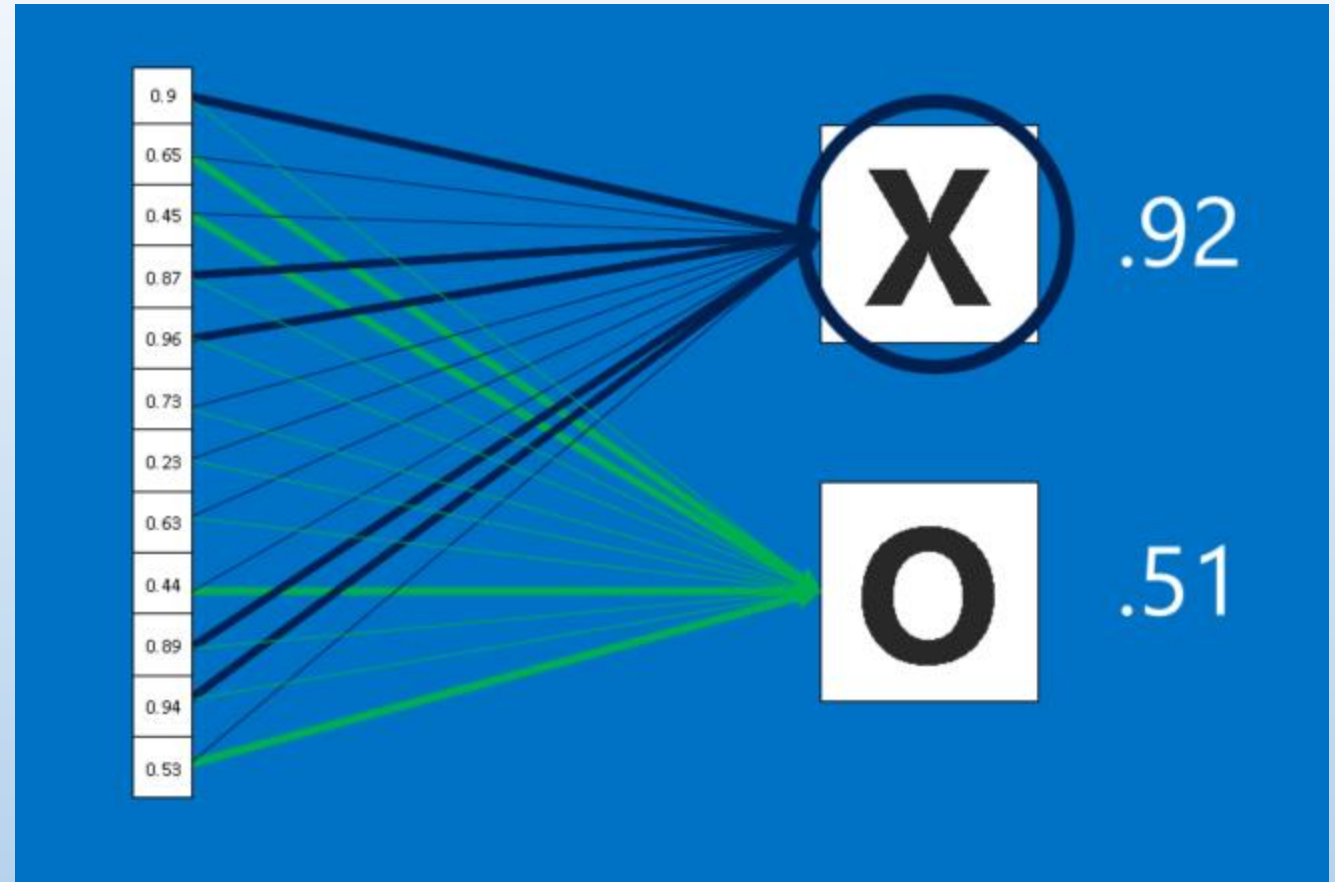
Les couches inférieures représentent les aspects simples de l'image, tels que les bords et points lumineux.

Les couches supérieures représentent des aspects beaucoup plus complexes de l'image, tels que des formes et des patterns.

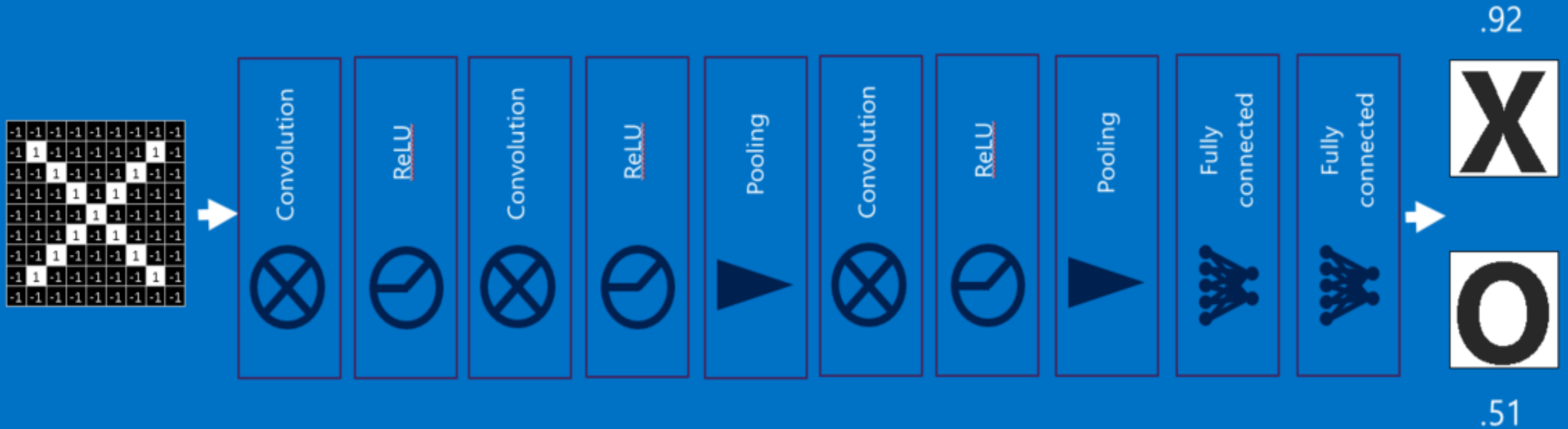
Couches entièrement connectées

Classifieur

*Les couches supérieures prennent en entrée les images de haut niveau.
Les tableaux de 2 dimensions sont considérés comme des listes et pondérés.
Les poids sont ajustés pour obtenir le résultat souhaité*



Réseau complet



Questions

- 1. D'où viennent les caractéristiques?*
- 2. Comment définit-on les poids de nos couches entièrement connectées?*

L' Apprentissage profond

Deep Learning

► Traditional Machine Learning



► Deep Learning



La rétro-propagation (Back propagation)



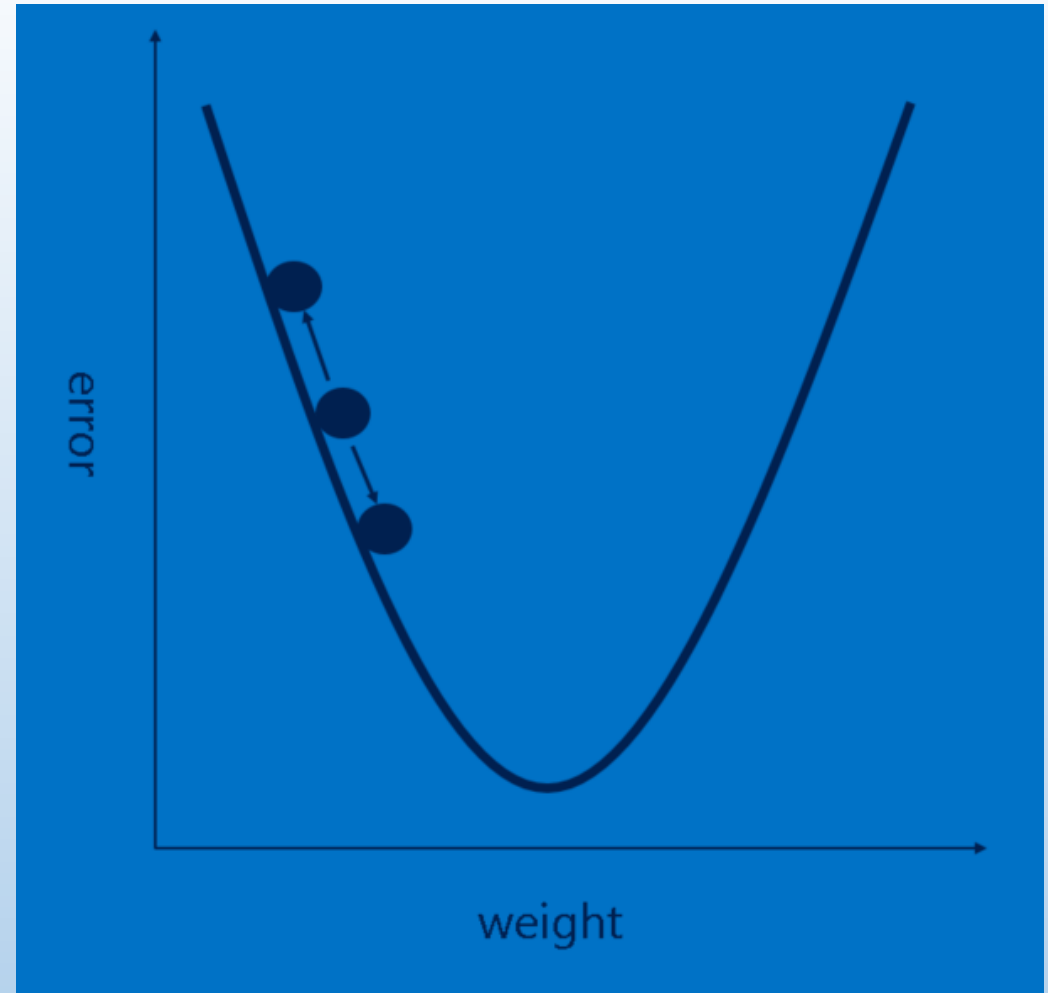
Au départ, chaque pixel de chaque caractéristique , ainsi que les poids de couches de classification sont fixés aléatoirement

Après ça, on commence à passer des images au CNN, une par une.

Les caractéristiques et poids sont ajustés pour obtenir le résultat souhaité

L'entraînement

*Les caractéristiques et poids peuvent ensuite être ajustés de manière à réduire l'erreur.
On soumet au CNN des images étiquetées
Si le résultat n'est pas celui attendu, chaque valeur est augmentée ou diminuée, et la nouvelle valeur de l'erreur de notre réseau est recalculée, à chaque fois.
Quel que soit l'ajustement fait, si l'erreur diminue, l'ajustement est conservé.
Ce processus est ensuite répété avec chacune des autres images qui ont un label.*



L'entraînement

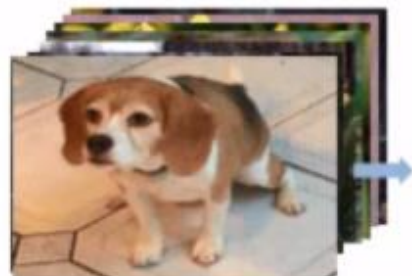
Les couches inférieures sont généralement entraînées à l'aide d'un apprentissage **non supervisé**, sans tâche de prédiction particulière à l'esprit, pour la détection des caractéristiques qui apparaissent fréquemment dans les données d'entrée.

Les couches supérieures sont toujours entraînées par des **techniques d'apprentissage automatique supervisé** telles que la **rétropropagation**.

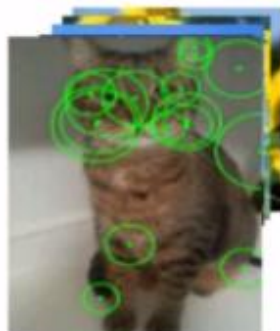
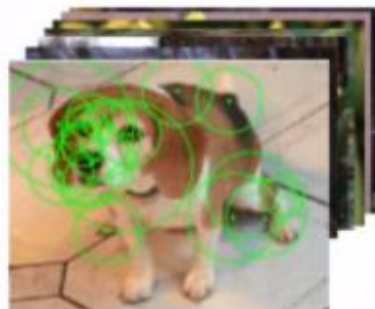
Si un mélange d'objets et de scènes est utilisé comme données d'entrée, les caractéristiques apprises par les couches inférieures seront plus ou moins génériques donc réutilisables.

Machine Learning Workflow

Training Data



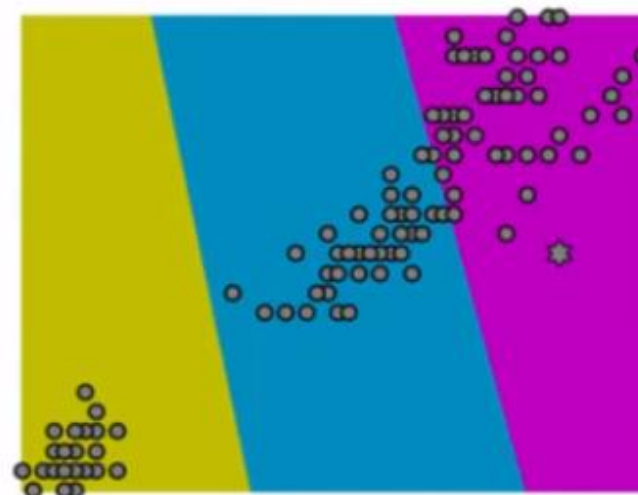
Feature Extraction



Test Data



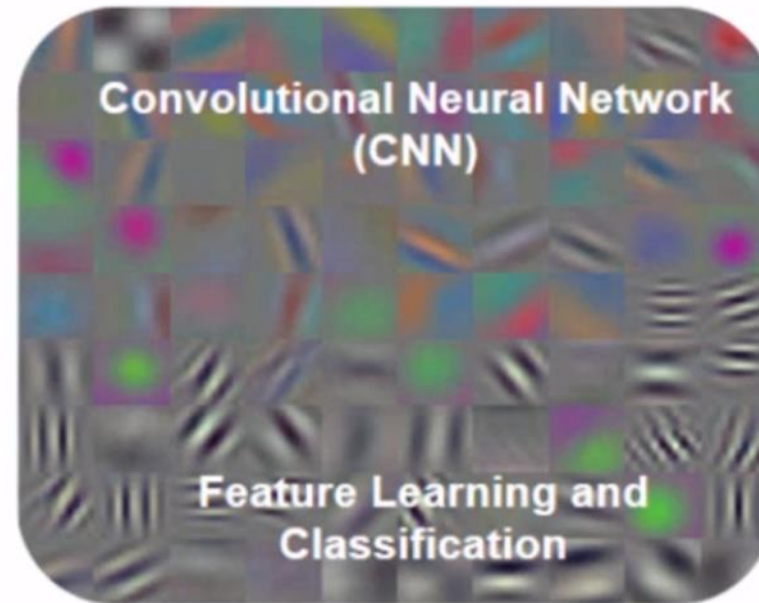
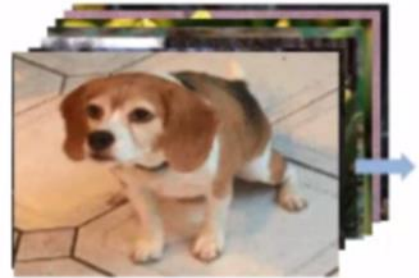
Machine Learning Model Classification



→ 'CAT'

Deep Learning Workflow

Training Data



Test Data

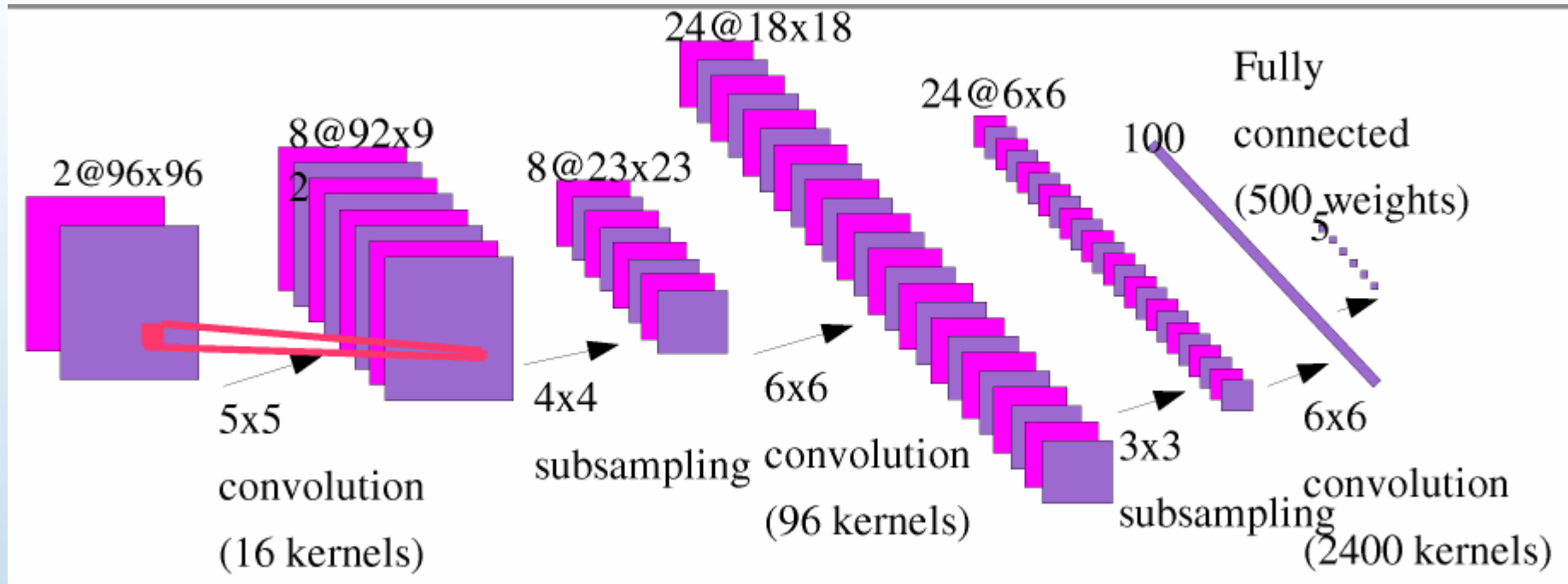


→ 'CAT'

LeNet de Yann Lecun



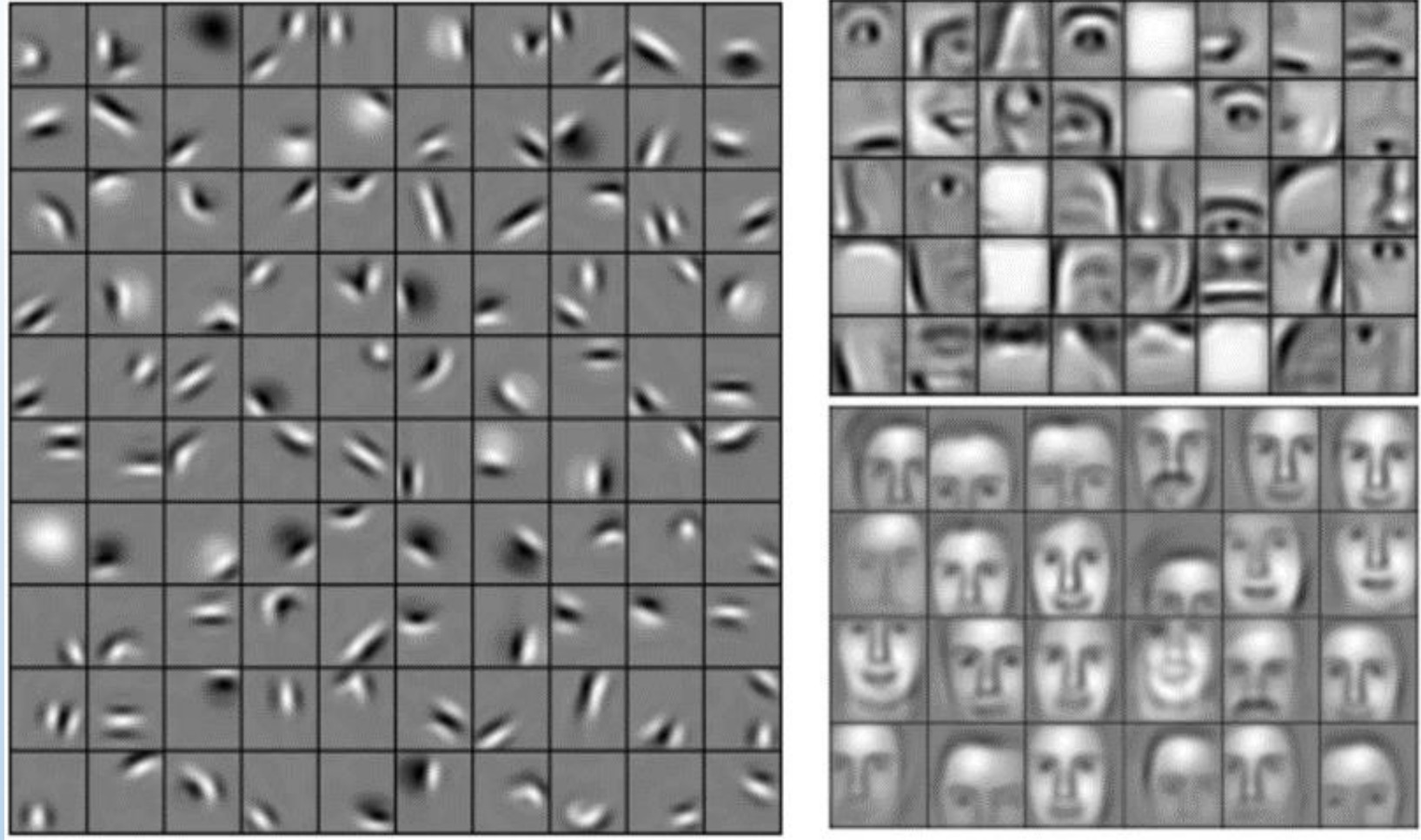
Yann LeCun



Architecture of the convolutional net "LeNet 7". This network has 90,857 trainable parameters and 4.66 Million connections. Each output unit is influenced by a receptive field of 96x96 pixels on the input.

Performances : <https://cs.nyu.edu/~yann/research/norb/>

L' Apprentissage profond



Dans un CNN entraîné à reconnaître des visages, les couches supérieures représentent des motifs et patterns qui font clairement partie d'un visage

Application à la reconnaissance des chiffres manuscrits

Sources:

Devon France (2017) : Martin Gorner

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



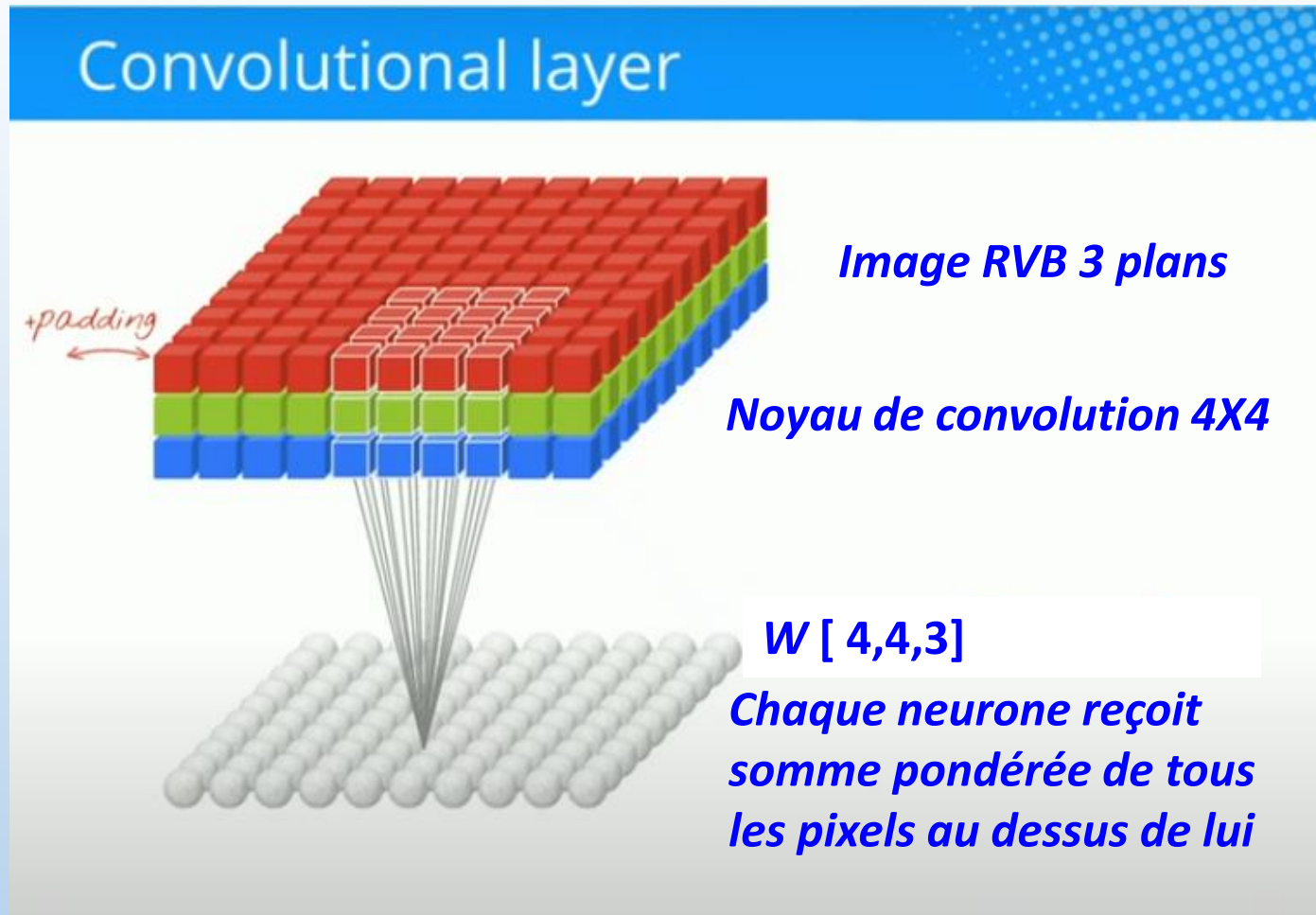
Martin Görner

Google Developer relations

[@martin_gorner](#)

plus.google.com/+MartinGorner

ConvNet

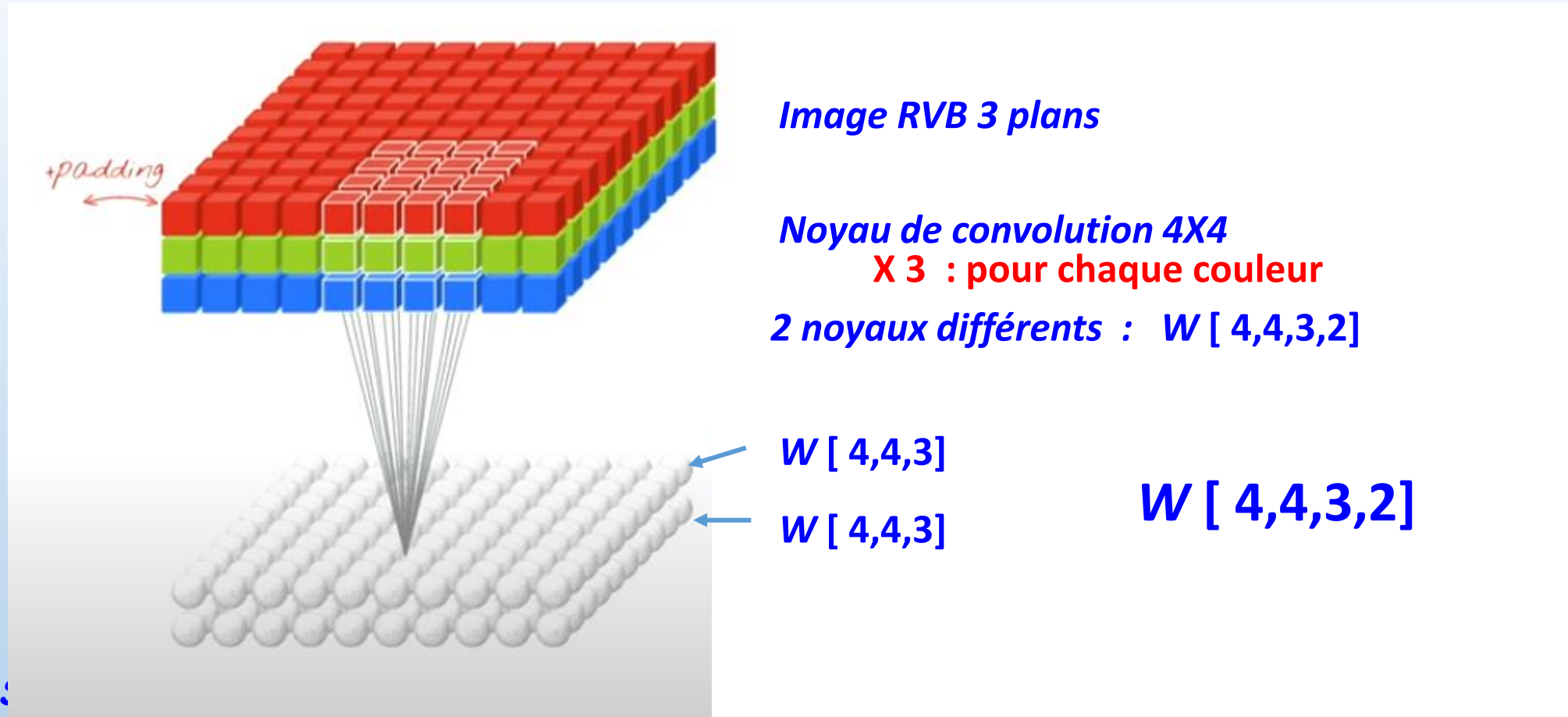


Sources :

Yann Lecun

Devox France : Martin Gorner <https://www.youtube.com/watch?v=BtAVBeLuigI>

ConvNet

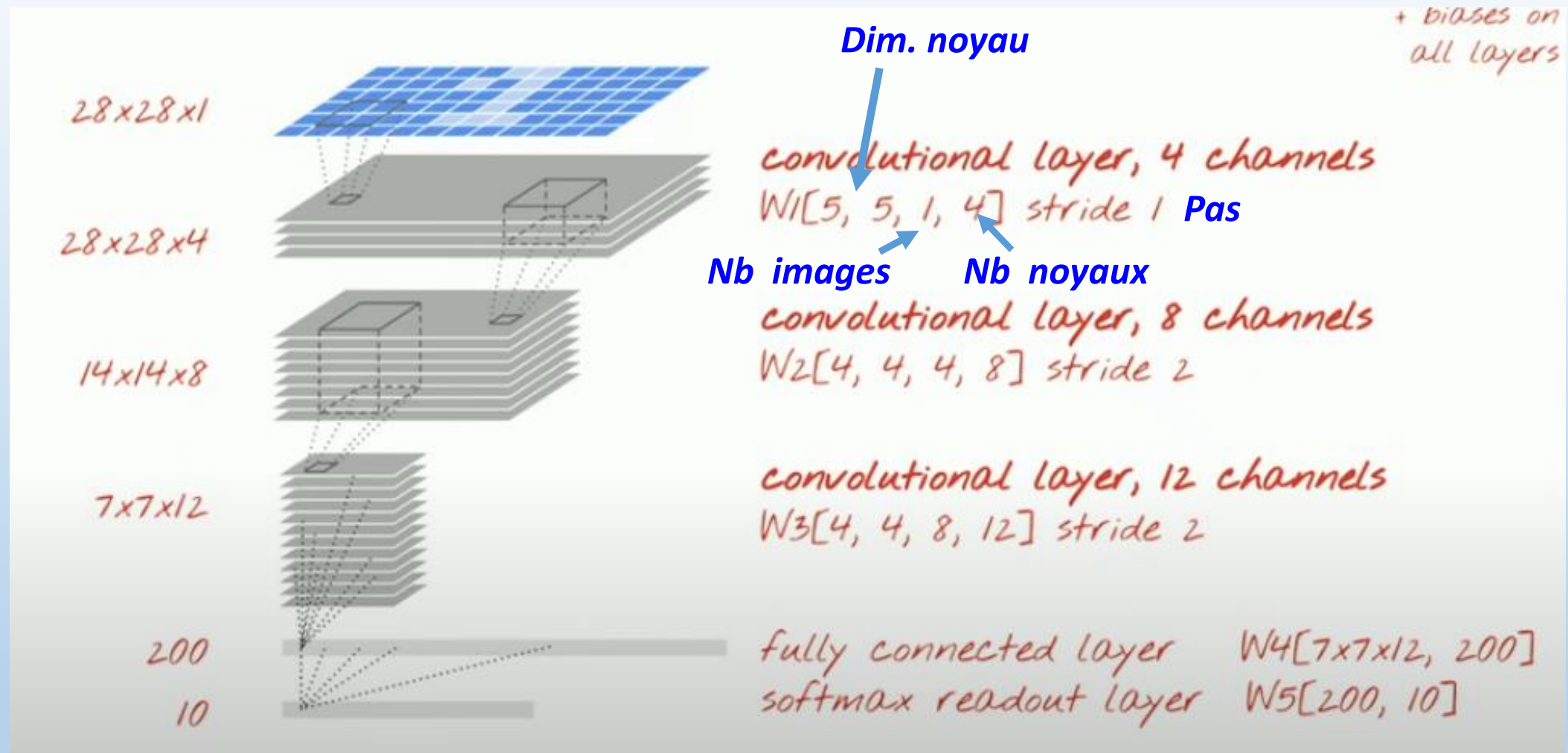


Yann Lecun

Devon France : Martin Gorner <https://www.youtube.com/watch?v=BtAVBeLuigI>

Reconnaissance des chiffres manuscrits

Première configuration



Reconnaissance des chiffres manuscrits

Première configuration

Tensorflow - initialisation

K=4
L=8
M=12

filter size *input channels* *output channels*

W1 = tf.Variable(tf.truncated_normal([5, 5, 1, K], stddev=0.1))

B1 = tf.Variable(tf.ones([K])/10)

W2 = tf.Variable(tf.truncated_normal([5, 5, K, L], stddev=0.1))

B2 = tf.Variable(tf.ones([L])/10)

W3 = tf.Variable(tf.truncated_normal([4, 4, L, M], stddev=0.1))

B3 = tf.Variable(tf.ones([M])/10)

N=200

*weights initialised
with random values*

W4 = tf.Variable(tf.truncated_normal([7*7*M, N], stddev=0.1))

B4 = tf.Variable(tf.ones([N])/10)

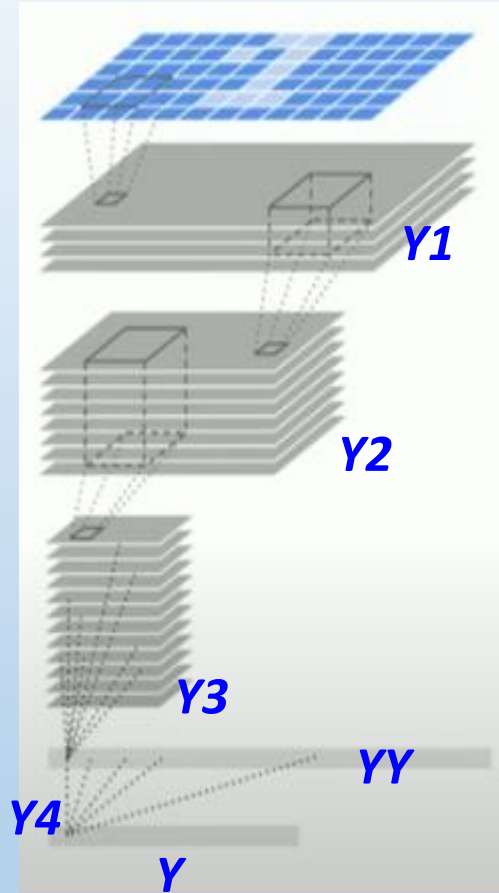
W5 = tf.Variable(tf.truncated_normal([N, 10], stddev=0.1))

B5 = tf.Variable(tf.zeros([10])/10)

Reconnaissance des chiffres manuscrits

Première configuration

Tensorflow - the model



input image batch
 $X[100, 28, 28, 1]$

weights

stride

biases

```
Y1 = tf.nn.relu(tf.nn.conv2d(X, W1, strides=[1, 1, 1, 1], padding='SAME') + B1)
Y2 = tf.nn.relu(tf.nn.conv2d(Y1, W2, strides=[1, 2, 2, 1], padding='SAME') + B2)
Y3 = tf.nn.relu(tf.nn.conv2d(Y2, W3, strides=[1, 2, 2, 1], padding='SAME') + B3)
```

```
YY = tf.reshape(Y3, shape=[-1, 7 * 7 * M])
```

```
Y4 = tf.nn.relu(tf.matmul(YY, W4) + B4)
```

```
Y = tf.nn.softmax(tf.matmul(Y4, W5) + B5)
```

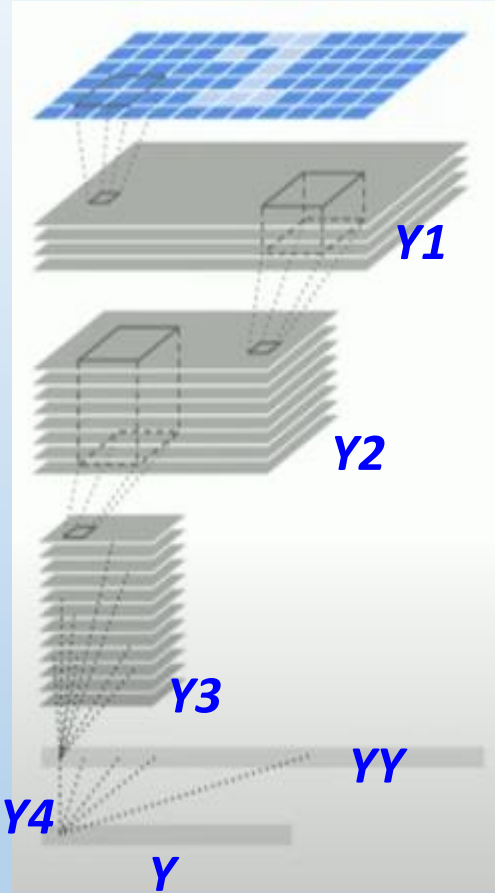
flatten all values for
fully connected layer

$Y3 [100, 7, 7, 12]$

$YY [100, 7 \times 7 \times 12]$

Reconnaissance des chiffres manuscrits

Première configuration

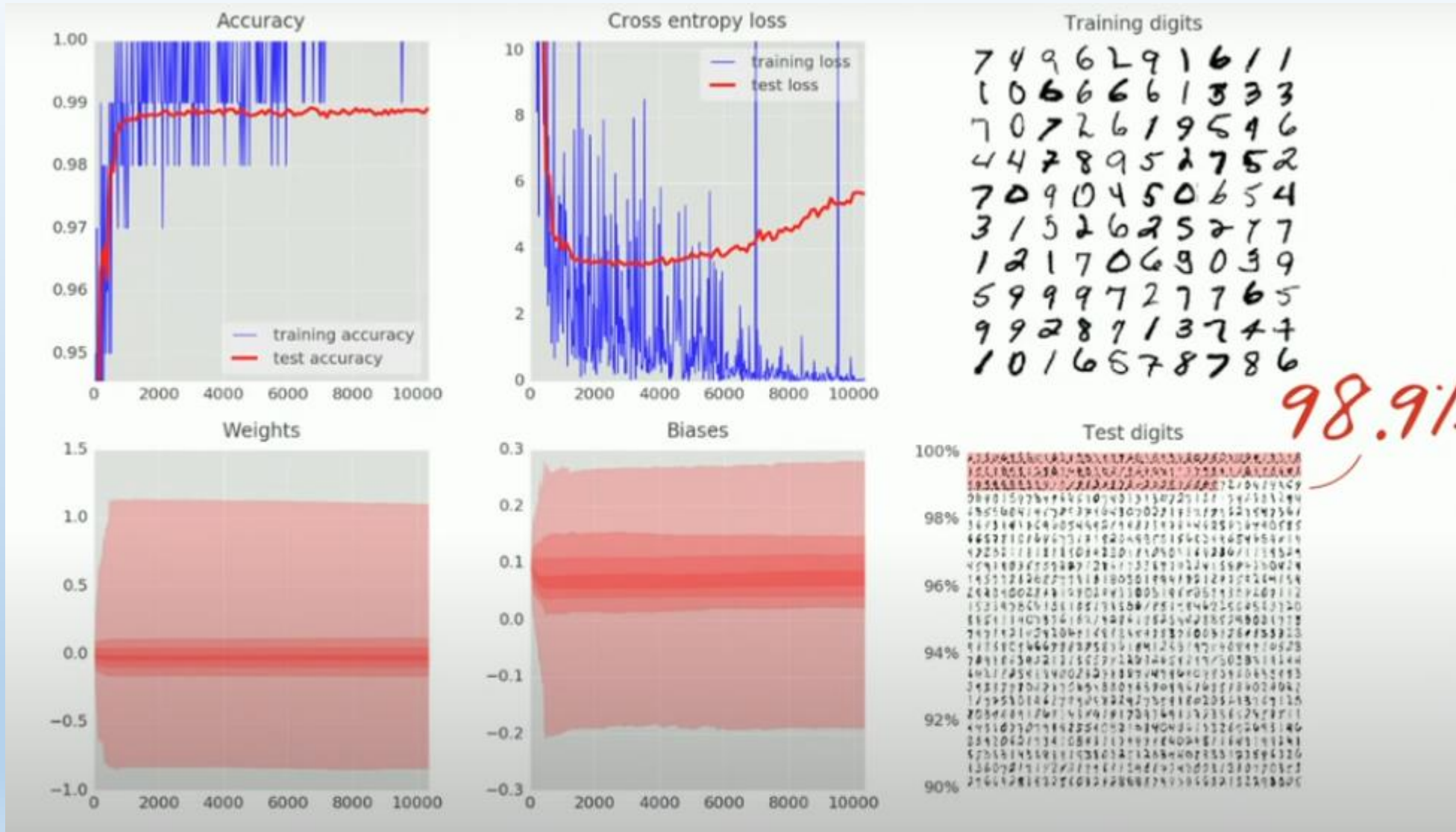


[Tensorflow et l'apprentissage profond, sans les équations différentielles \(Martin Görner\) \(youtube.com\)](#)

Séquence entraînement début 1 h 01 '21'''

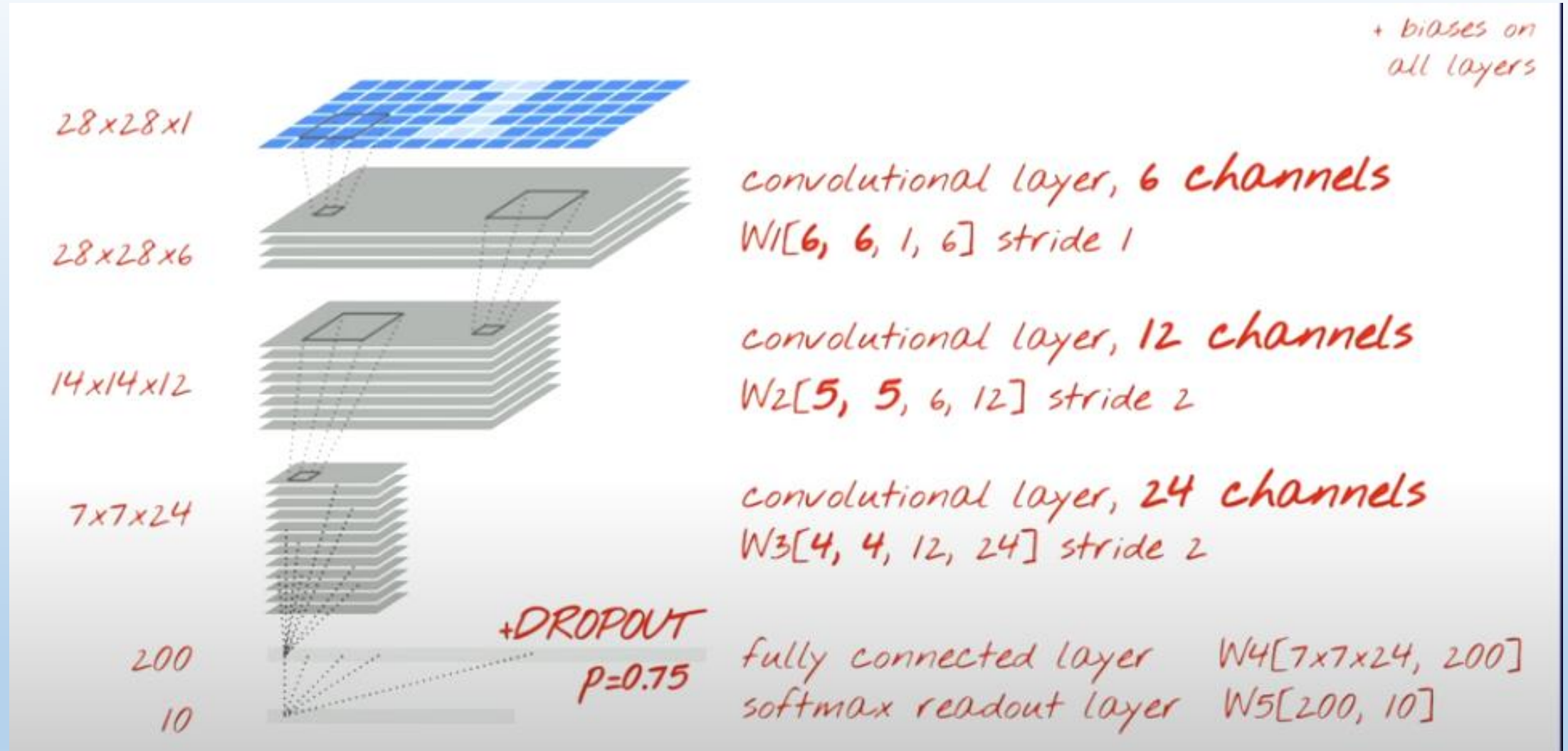
Reconnaissance des chiffres manuscrits

Première configuration



Reconnaissance des chiffres manuscrits

Deuxième configuration



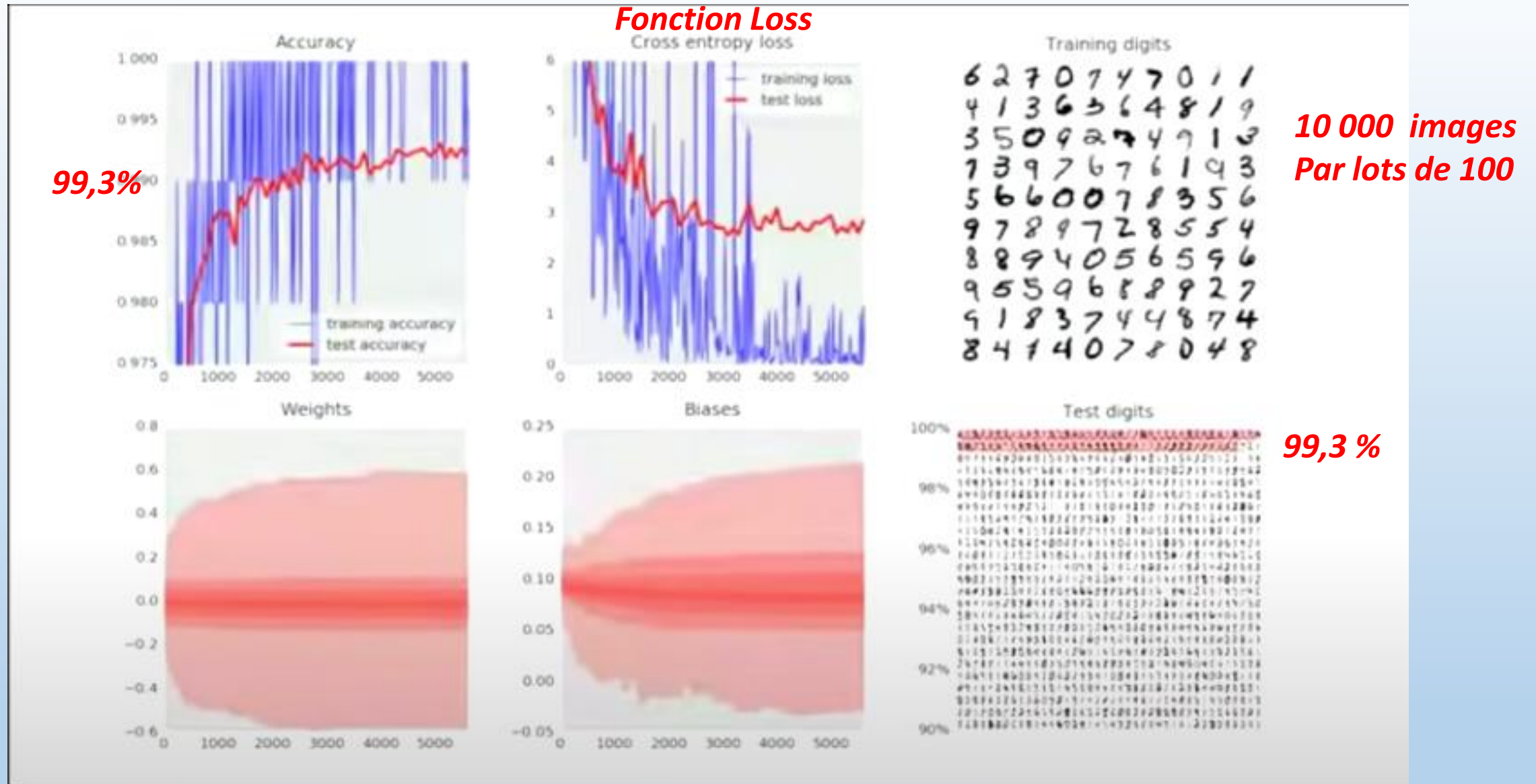
Entraînement

Source Devovx France

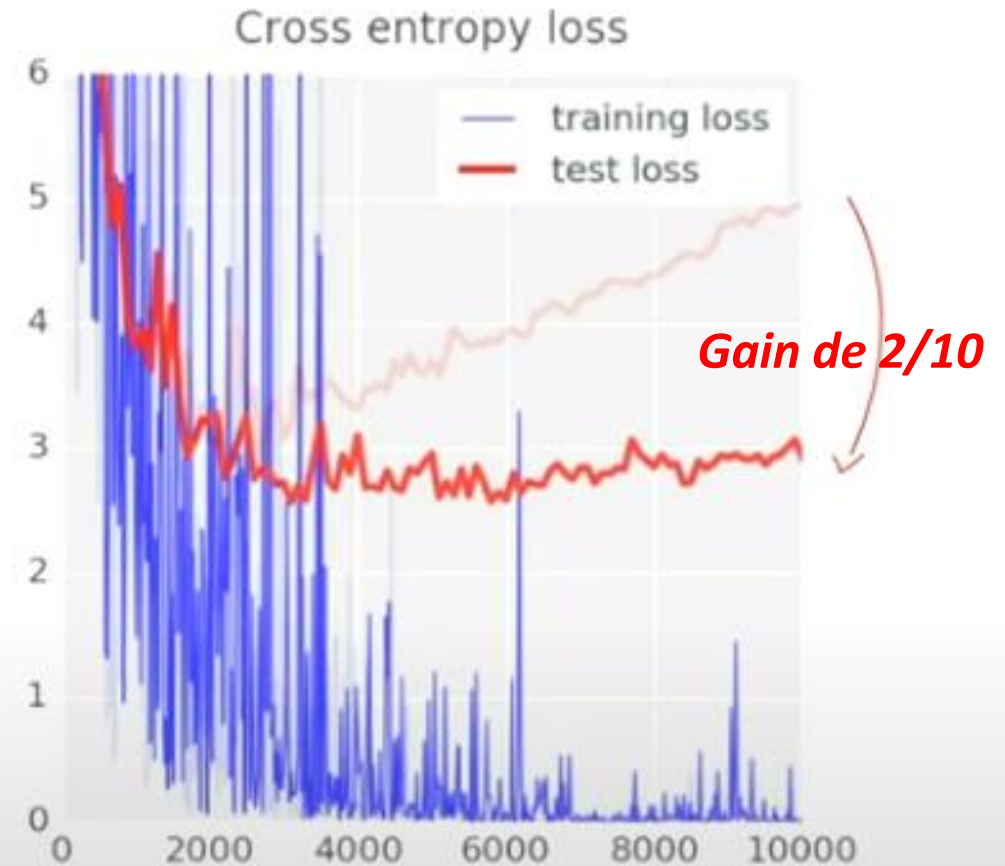
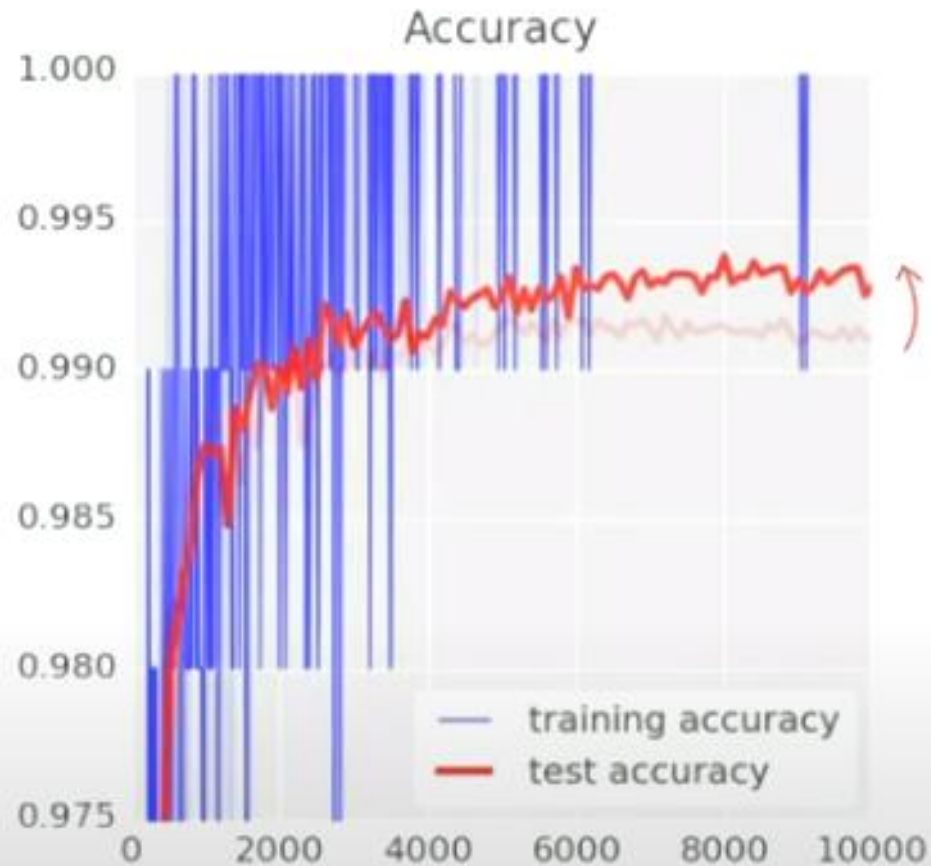
[Tensorflow et l'apprentissage profond, sans les équations différentielles \(Martin Görner\) \(youtube.com\)](#)

Séquence entraînement début 1h 05'

Reconnaissance des chiffres manuscrits

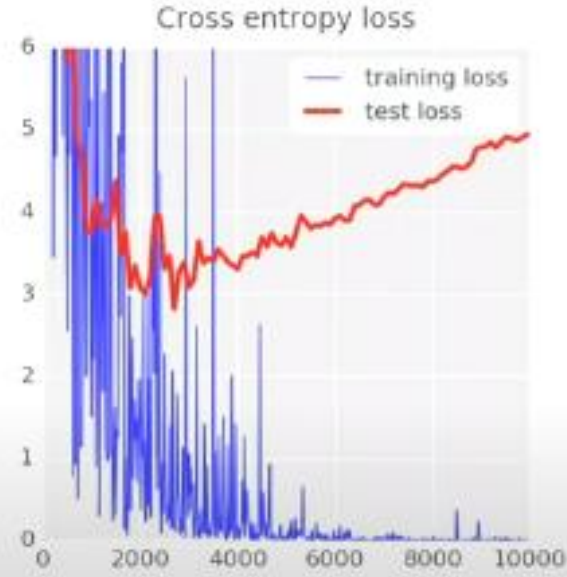


Reconnaissance des chiffres manuscrits

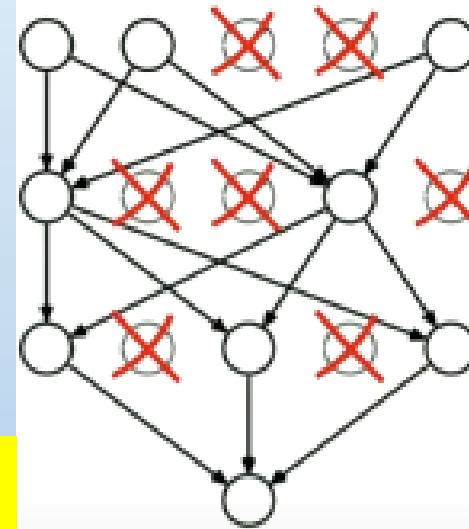


Avec régularisation Dropout

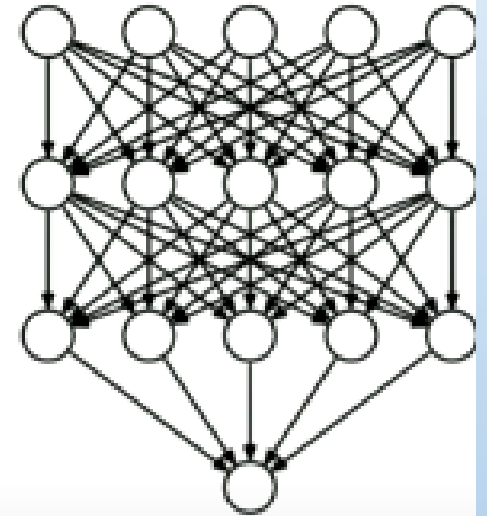
Régularisation Dropout



Overfitting : sur apprentissage
Trop de neurones



TRAINING
 $p_{\text{keep}}=0.75$

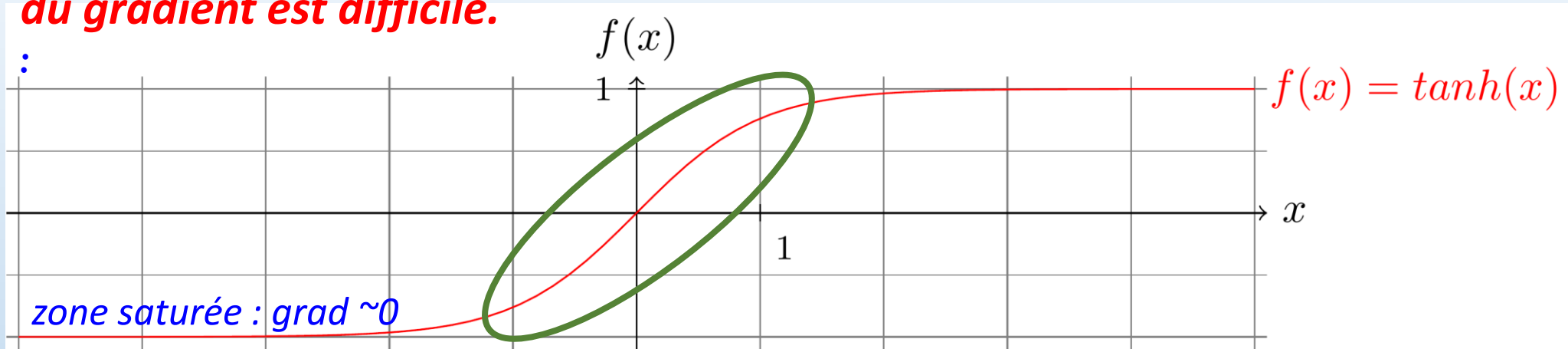


EVALUATION
 $p_{\text{keep}}=1$

Dropout

Gradient évanescent Normalisation par lots

Plus on rajoute de couches, plus l'apprentissage par rétropropagation du gradient est difficile.



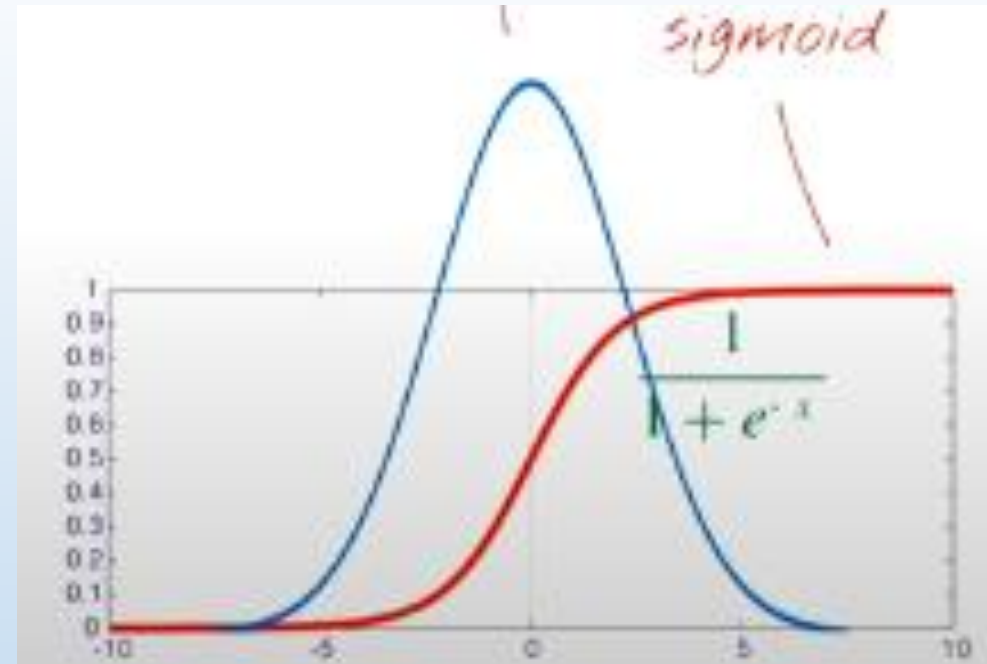
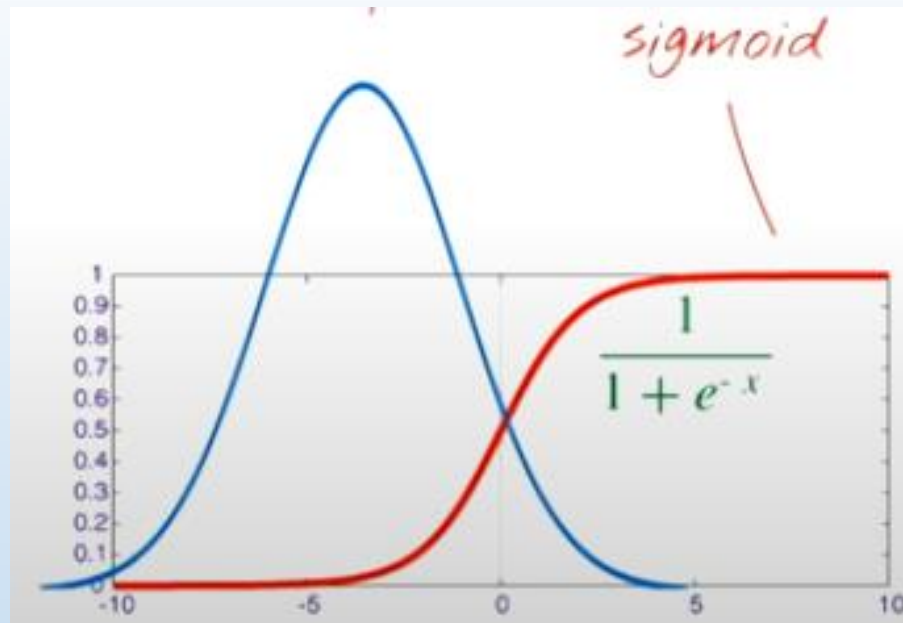
→ Une réponse : Couche Batch Normalization

Réduction du phénomène d'évanescence du gradient

Permet de maintenir les exemples d'un batch (lot) dans la zone non saturée d'une unité.

Par exemple, la zone proche de 0 dans la fonction tangente hyperbolique

Normalisation par lots



Couche Batch Normalization

Pour les sorties de chaque couche de neurones :

On calcule la moyenne et les écarts-types d'un lot

On fait l'opération suivante pour chaque sortie :

Valeur sortie – valeur moyenne

Écart type

Applicable aux **sorties pondérées** avant la fonction d'activation

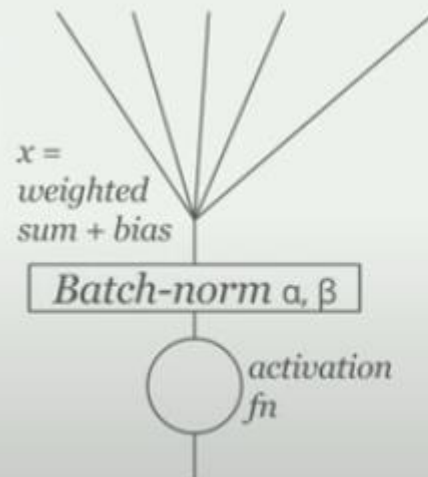
Normalisation par lots

Batch normalisation

depends from:
weights, biases, images

depends from:
same weights and biases, images
only one set of weights and biases in a mini-batch

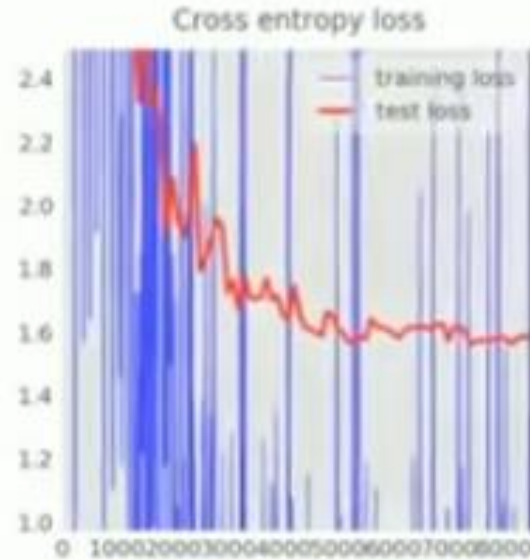
$$\hat{x} = \frac{x - avg_{batch}(x)}{stdev_{batch}(x) + \epsilon}$$



$$BN(x) = \alpha \hat{x} + \beta$$

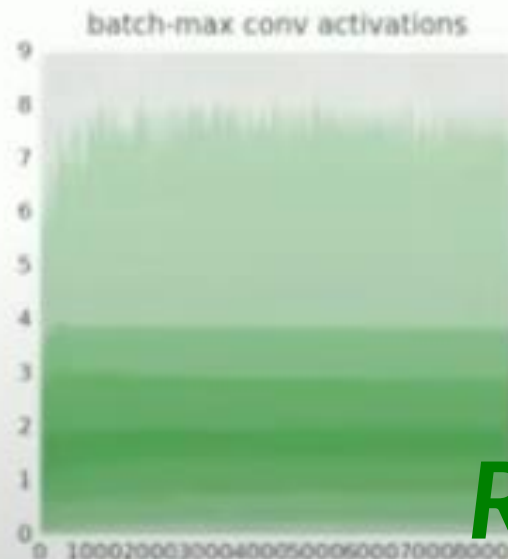
=> BN is differentiable relatively to weights, biases, α and β
It can be used as a layer in the network, gradient calculations will still work

Avec Dropout et BatchNorm



Training digits

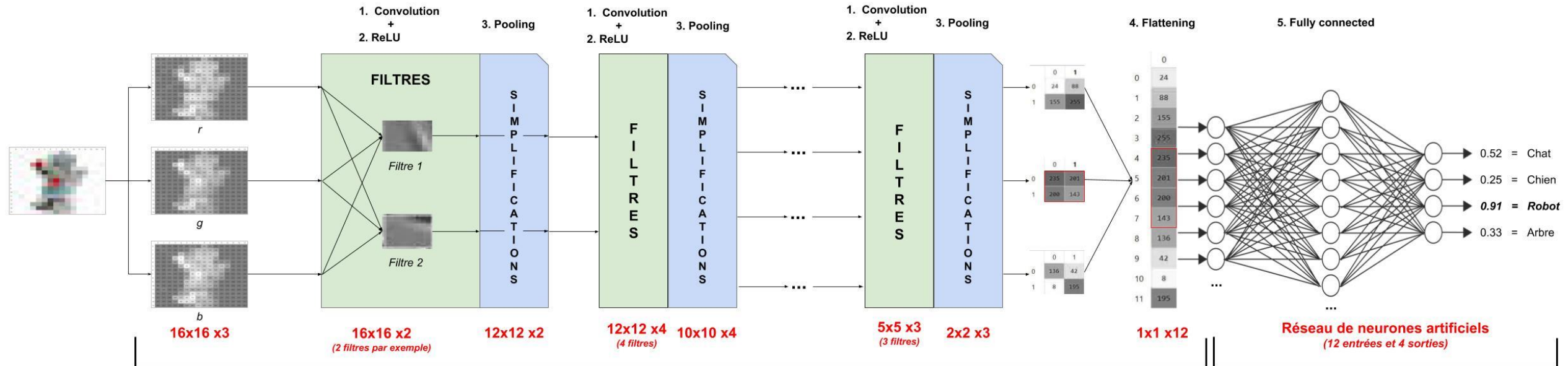
3	0	8	8	5	7	8	4	2	1
9	1	3	3	8	9	9	2	1	
4	3	1	4	0	6	7	6	7	1
0	8	1	9	1	3	4	4	3	3
0	0	3	3	9	2	2	7	3	1
7	7	7	4	5	3	6	0	8	0
7	1	3	0	0	6	1	9	0	0
4	1	3	0	5	0	4	2	0	7
0	5	7	4	0	9	7	0	8	4
5	0	7	9	6	5	7	1	9	9



99,5 %

Record du monde 99,7 %

Les CNN ou ConvNet



Extraction des informations de l'image grâce à un enchaînement de filtres (et de simplifications)

Prédiction de la classe de l'image

Reconnaissance images

Explicateur CNN

<https://poloclub.github.io/cnn-explainer/>

Reconnaissance images

Tester Expérimenter

Source INRIA

<https://pixees.fr/classcode-v2/>

Ouvrir un compte et se connecter

Puis ouvrir **Class'code IAI**

The screenshot shows the 'Class'Code IAI' interface. At the top, there's a navigation bar with a hamburger menu, '<Class'Code>', and a user profile 'CHRIS49'. Below this is a purple banner with the 'Class'Code IAI' logo and three featured sections: '#1 VOUS AVEZ DIT IA?', '#2 BOOSTÉ À L'IA!', and '#3 HUMAINS ET IA...'. On the left, a sidebar contains four buttons: 'Se questionner', 'Expérimenter' (highlighted in red), 'Découvrir', and 'Débattre'. The main content area is titled 'TESTONS NOTRE PREMIER PROGRAMME' in red. It features a video player with a thumbnail showing a person and the text 'VOUS AVEZ DIT IA?'. Below the video, there's a paragraph of text: 'C'est quoi l'IA et ça fonctionne comment un programme d'IA ? On vous explique que ça n'a rien de magique ; et d'ailleurs allez-y, testez en entraînant une IA !'. The video player has a play button and a 'SUITE >' button.

Au-delà des images

Quel que soit le type de données nous pouvons transformer ces données pour les faire ressembler à une image → Vecteurs .

Les signaux audio peuvent être décomposés en un ensemble de petits morceaux :

→ tableau de 2 dimensions : colonne = bloc de temps, ligne = bande de fréquences

*De même pour les données **textuelles** pour le traitement du langage naturel et même des **données chimiques** pour la découverte de médicaments*

Les CNNs fonctionnent très bien dans ces cas précis.

L' Apprentissage profond

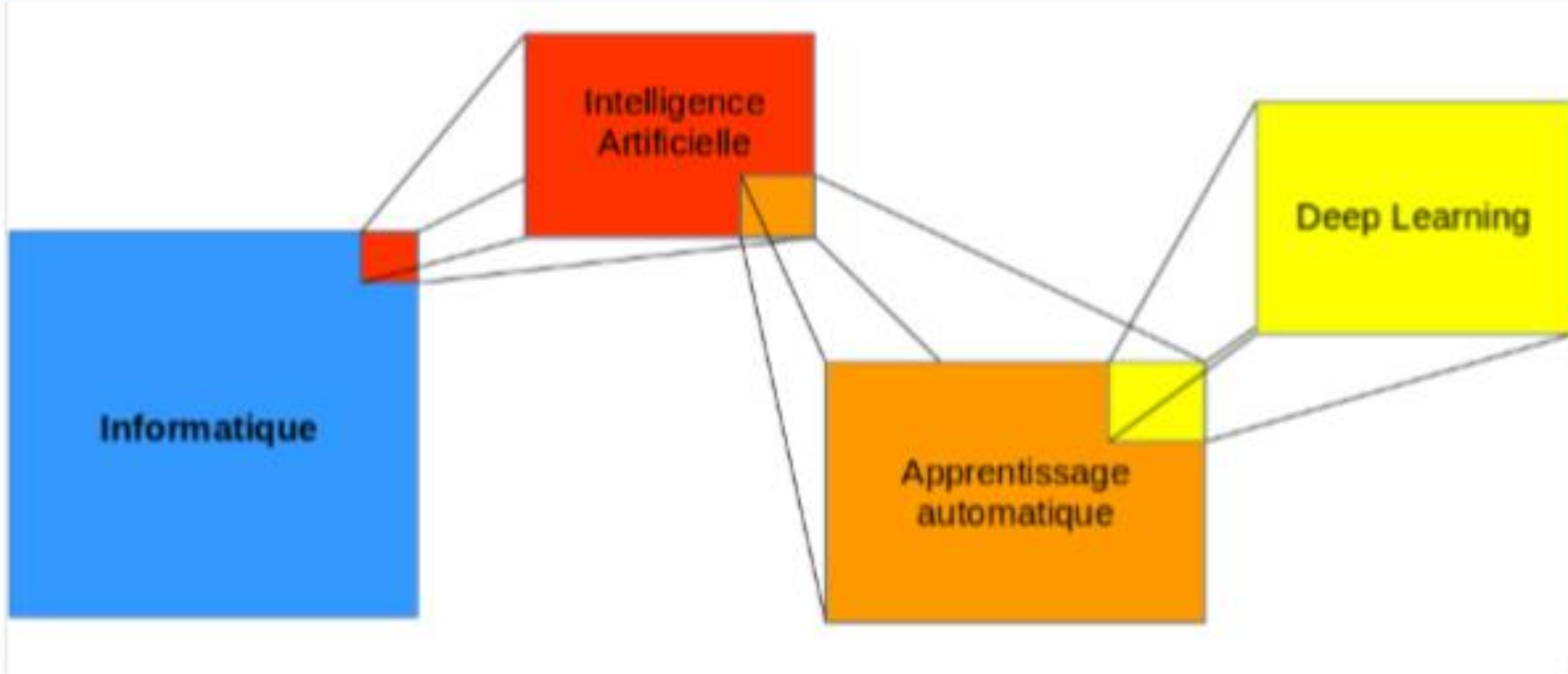


Figure 1: Schéma résumant la place du deep learning dans le monde de l'informatique.